

Docker on Openstack

http://www.slideshare.net/osc_hojinkim/docker-on-openstack-by-osc



2015. 3. 12

(주) 오픈소스컨설팅

김호진

Da IT

최신기사

실시간속보

한국의 리

OPEN O

삼성 갤럭시S6 CL

삼성디스플레이, 투

애플, '애플워치' O

IBM

pSer

Ser

SA38

Cha

Mail

Sys

S

Chw

FRL

Sen

Use

Sen

C

T

S

V

A

Acc

A

A

Res

P

P

Chw

Ent

Out

C

MA

P

MA

P

IBM PowerVC

What is New

- PowerVP and mobile CoD are explained
- Shared Storage Pool enhancements explained
- Power Integrated Facility for Linux described

ibm.com/redbooks

IBM PowerVC

Introduction and Configuration



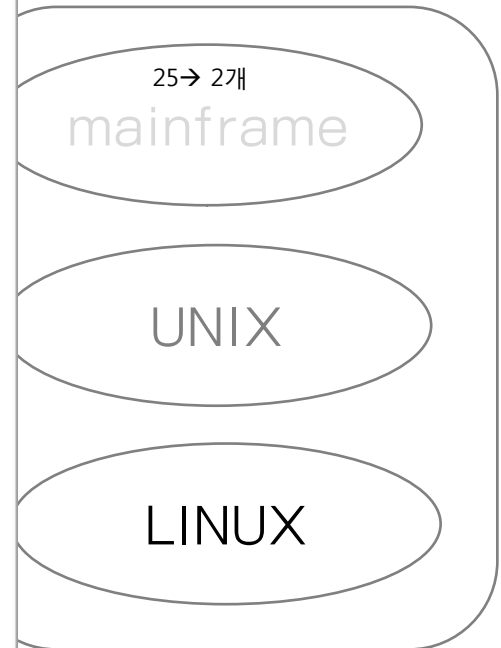
- OpenStack compatibility for integration with cloud software stacks
- Integration of server and storage virtualization
- IBM PowerVM virtualization

ibm.com/redbooks



Bruno Blanchard
Guillermo Corti
Sylvain Delabarre
Ho Jin Kim
Ondrej Plachy
Marcos Quezada
Gustavo Santos

Redbooks



Contents

1. **Openstack 인프라 구축 (4 node 구성) [30분]**
2. Openstack 위에 VM 생성 [20분]
3. docker 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. Docker로 WEB서비스 구축 [30분]
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. Docker로 jenkins 구현 [15분]

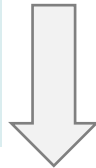
왜 Docker on Openstack 인가?

OpenStack / KVM / Docker

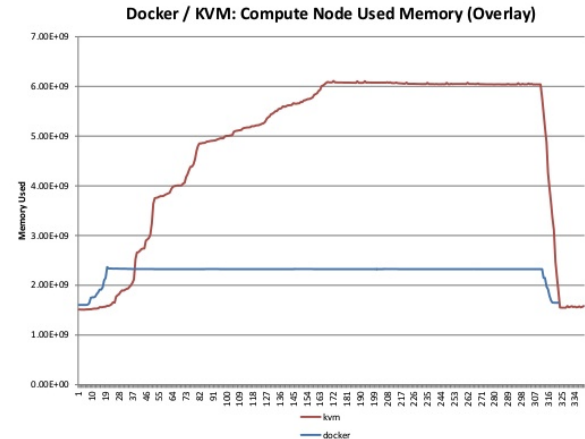
- Openstack은 전반적인 datacenter 운영
- KVM 기반 가상화는 컴퓨터 자원관리 측면
- Docker는 어플리케이션 배포관련 컨테이너

• Openstack은 클라우드 인프라 스트럭처에서 제공해 주는 멀티테넌트의 보안 및 격리, 관리 및 모니터링, 스토리지 및 네트워킹등은 전반적인 datacenter 운영 기반

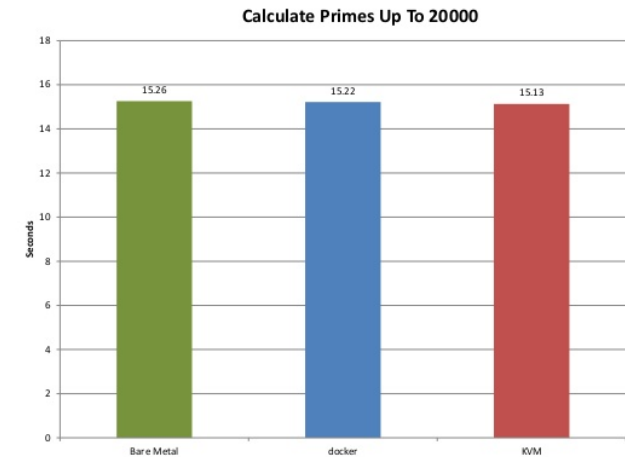
- Docker는 높은 이식성, 하드웨어, Framework 독립적인 컨테이너.



속도 / 효율성/ 이동성
 더 적은 메모리/CPU OVERHEAD
 Kvm/vmwaere/virtual machine 어디든
 도커라는것을 인식하지 못한채 리눅스 컨테이너 관리가능



오픈스택위에 리소스 개수에 따른 메모리 사용률

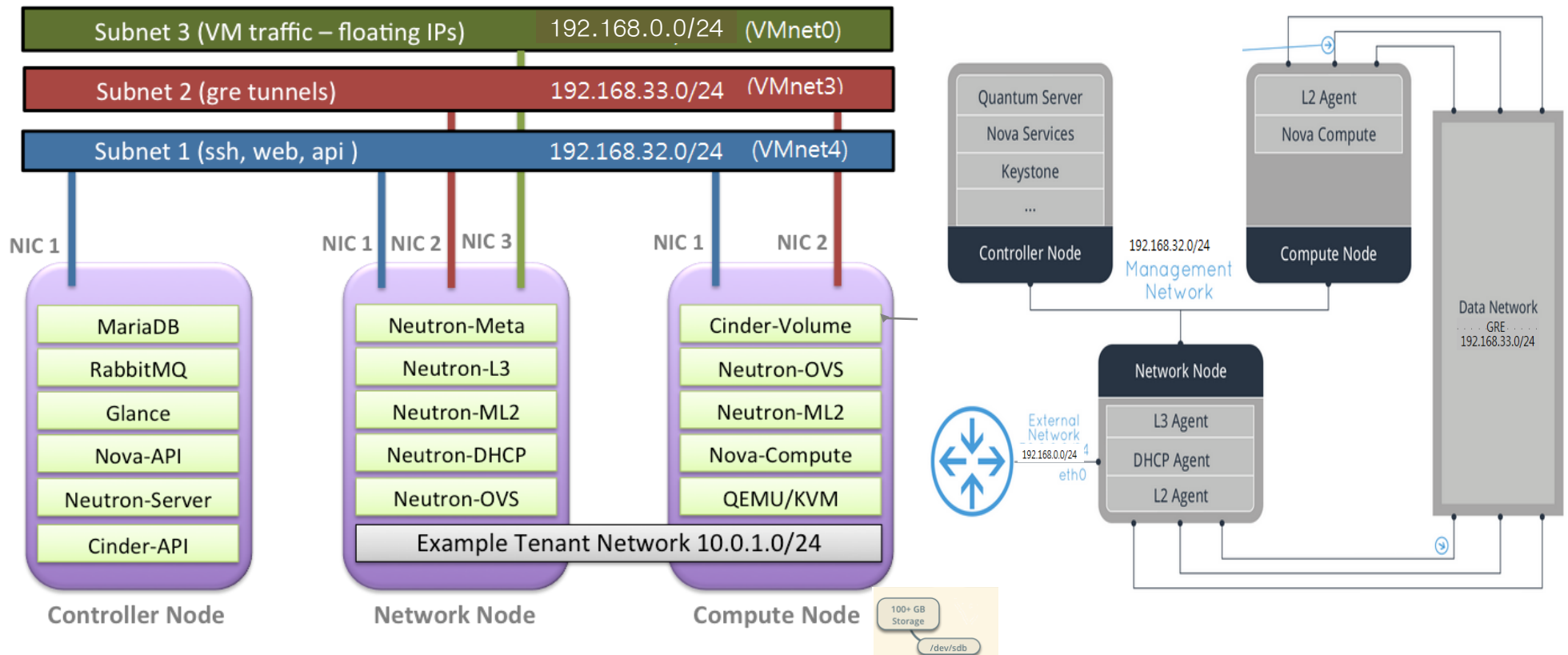


성능비교표

What to do during 30 min.

OpenStack Juno Install with Neutron on CentOS 7

- 3 nodes configuration (default)
- 3 networks configuration (tunnel=>gre)
- Local cinder (limits of Test Bed), but disk was divided to another disk (uses alone)



What to do during 30 min.

OpenStack Juno Install with Neutron on CentOS 7

- 후에 최종적으로 4 node configuration으로 변환될 예정임. (+1 compute node)

ENV
CONTROLLER_IP=192.168.32.181
ADMIN_TOKEN=ADMIN
SERVICE_PWD=service
ADMIN_PWD=password
META_PWD=meta123

#juno-controller node
THISHOST_NAME=juno-controller
THISHOST_IP=192.168.32.181
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=na
THISHOST_TUNNEL_NETMASK=255.255.255.0

#juno-network node
THISHOST_NAME=juno-network
THISHOST_IP=192.168.32.182
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=192.168.33.182
THISHOST_TUNNEL_NETMASK=255.255.255.0

#juno-compute node
THISHOST_NAME=juno-compute
THISHOST_IP=192.168.32.184
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=192.168.33.184
THISHOST_TUNNEL_NETMASK=255.255.255.0

#juno-compute node
THISHOST_NAME=juno-compute
THISHOST_IP=192.168.32.183
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=192.168.33.183
THISHOST_TUNNEL_NETMASK=255.255.255.0

첫번째 eth0에는 각기 아래 ip가 부여됩니다. (Mgt network)

- juno-controller: 192.168.32.181 / juno-network: 192.168.32.182
- juno-compute01: 192.168.32.183 / juno-compute02: 192.168.32.184

두번째 eth1에는 각기 아래 ip가 부여됩니다. (tunneling network)

- * juno-network: 192.168.33.182 / juno-compute01: 192.168.33.183 / juno-compute02: 192.168.33.184

세번째 eth2에는 floating IP를 가지게 됩니다. (public network-floating)

- * juno-network: public IP는 따로 IP를 주지 않음.

openstack 구축시 log 보는법

conf 파일에 verbose=true 옵션을 걸어 더 상세히 볼수 있음.

- openstack의 대부분 내용은 /var/log/messages에 나옴.
- verbose=true option을 주면 /var/log/messages가 각자 원하는 /var/#service/#service.log가 나옴 .
- 그리고 /var/log/messages에 error뿐 아니라 동작 내용까지 모두 기록됨.

```
기존 log ( /var/log/cinder/scheduler.log)
2015-03-13 03:09:12.360 1148 INFO oslo.messaging._drivers.impl_rabbit [req-844f54aa-6201-4fc4-b321-c6ab2012c296 - -
- - -] Connecting to AMQP server on 192.168.32.181:5672
2015-03-13 03:09:12.433 1148 ERROR oslo.messaging._drivers.impl_rabbit [req-844f54aa-6201-4fc4-b321-c6ab2012c296 -
- - -] AMQP server on 192.168.32.181:5672 is unreachable: [Errno 111] ECONNREFUSED. Trying again in 3 seconds.

verbose=true option 적용시 (/var/log/cinder/scheduler.log)

2015-03-13 06:20:18.812 18581 INFO cinder.service [-] Starting cinder-scheduler node (version 2014.2.1)
2015-03-13 06:20:18.816 18581 INFO oslo.messaging._drivers.impl_rabbit [req-1d1a9b9c-3658-4f76-8dc1-3d74b2028a36 -
- - -] Connecting to AMQP server on 192.168.32.181:5672
2015-03-13 06:20:18.837 18581 INFO oslo.messaging._drivers.impl_rabbit [req-1d1a9b9c-3658-4f76-8dc1-3d74b2028a36 -
- - -] Connected to AMQP server on 192.168.32.181:5672
2015-03-13 06:20:19.291 18581 INFO oslo.messaging._drivers.impl_rabbit [-] Connecting to AMQP server on
192.168.32.181:5672
2015-03-13 06:20:19.303 18581 INFO oslo.messaging._drivers.impl_rabbit [-] Connected to AMQP server on
192.168.32.181:5672
2015-03-13 06:20:50.814 18581 WARNING cinder.scheduler.host_manager [req-00223525-0c03-4c5d-ae9b-690ae0a10e72
d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] volume service is down. (host: jun-
compute)
2015-03-13 06:20:50.814 18581 WARNING cinder.scheduler.filter_scheduler [req-00223525-0c03-4c5d-ae9b-690ae0a10e72
d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] No weighed hosts found for volume with
properties: {}
2015-03-13 06:20:50.816 18581 ERROR cinder.scheduler.flows.create_volume [req-00223525-0c03-4c5d-ae9b-690ae0a10e72
d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] Failed to run task
cinder.scheduler.flows.create_volume.ScheduleCreateVolumeTask;volume:create: No valid host was found. No weighed
hosts available
```

openstack 구축시 log 보는법

conf 파일에 verbose=true 옵션을 걸어 더 상세히 볼수 있음.

```
verbose=true option 적용시 (/var/log/messages)
```

```
Mar 13 06:20:50 jun0-controller cinder-api: 2015-03-13 06:20:50.230 18615 INFO cinder.api.v1.volumes [req-00223525-0c03-4c5d-ae9b-690ae0a10e72 d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] Create volume of 2 GB
Mar 13 06:20:50 jun0-controller cinder-api: 2015-03-13 06:20:50.620 18615 INFO oslo.messaging.drivers.impl_rabbit [req-00223525-0c03-4c5d-ae9b-690ae0a10e72 d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] Connecting to AMQP server on 192.168.32.181:5672
Mar 13 06:20:50 jun0-controller cinder-api: 2015-03-13 06:20:50.643 18615 INFO oslo.messaging.drivers.impl_rabbit [req-00223525-0c03-4c5d-ae9b-690ae0a10e72 d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] Connected to AMQP server on 192.168.32.181:5672
Mar 13 06:20:50 jun0-controller cinder-api: 2015-03-13 06:20:50.686 18615 INFO cinder.api.v1.volumes [req-00223525-0c03-4c5d-ae9b-690ae0a10e72 d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] vol={'migration_status': None, 'availability_zone': 'nova', 'terminated_at': None, 'reservations': ['01680237-32b1-4bcb-a3d4-c17b46837298', 'dd9280a1-7232-4aba-acf0-23aef02c34a9'], 'updated_at': None, 'provider_geometry': None, 'replication_extended_status': None, 'replication_status': 'disabled', 'snapshot_id': None, 'ec2_id': None, 'mountpoint': None, 'deleted_at': None, 'id': '37d5a43a-3f6c-4880-91c6-7fba7c434211', 'size': 2, 'user_id': u'd13d86ad609d4a9a8d9a84b36b954a69', 'attach_time': None, 'source_replicaid': None, 'attached_host': None, 'display_description': None, 'volume_admin_metadata': [], 'project_id': u'3c402245243f443ebc2aa39605641be1', 'launched_at': None, 'scheduled_at': None, 'status': 'creating', 'volume_type_id': None, 'deleted': False, 'provider_location': None, 'host': None, 'consistencygroup_id': None, 'source_valid': None, 'provider_auth': None, 'display_name': u'test2', 'instance_uuid': None, 'bootable': False, 'created_at': datetime.datetime(2015, 3, 13, 10, 20, 50, 562087), 'attach_status': 'detached', 'volume_type': None, 'consistencygroup': None, 'volume_metadata': [], '_name_id': None, 'encryption_key_id': None, 'replication_driver_data': None, 'metadata': {}}
Mar 13 06:20:50 jun0-controller cinder-api: 2015-03-13 06:20:50.700 18615 INFO cinder.api.openstack.wsgi [req-00223525-0c03-4c5d-ae9b-690ae0a10e72 d13d86ad609d4a9a8d9a84b36b954a69 3c402245243f443ebc2aa39605641be1 - - -] http://192.168.32.181:8776/v1/3c402245243f443ebc2aa39605641be1/volumes returned with HTTP 200
```


Env setting

사전 환경 조성 (all node에서 실행)

- Ethernet name을 익숙한 eth#으로 변경한다.

```
cat /etc/default/grub
```

```
[root@juno-controller ~]# cat /etc/default/grub
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=centos/swap vconsole.font=latarcyrheb-sun16 rd.lvm.lv=centos/root
crashkernel=auto vconsole.keymap=us net.ifnames=0 rhgb quiet"
GRUB_DISABLE_RECOVERY="true"
```

```
[root@juno-controller ~]# sudo gurb2-mkconfig -o /boot/grub2/grub.cfg
```

- NetworkManger stop (추가 설치 : yum install -y net-tools)

```
systemctl stop NetworkManager
systemctl disable NetworkManager
```

- NTP설정 (yum install ntp) → /etc/ntp.conf

```
서버 : controller 서버에 자기 정보 기입
restrict 192.168.32.0 mask 255.255.255.0 nomodify
notrap
server 127.127.1.0 iburst # local clock
```

```
나머지 2 node ( network/compute node)
server 192.168.32.181 iburst # local clock
```

```
systemctl start ntpd.service
systemctl enable ntpd.service
```

Env setting

환경 변수 setting (env parameter setting)

- OpenStack용 rpm을 제공하는 Third Party Repository를 모든 노드에 설정

All node

```
Centos 7 minimum installation
yum -y install epel-release
yum -y install http://rdo.fedorapeople.org/openstack-juno/rdo-release-juno.rpm
yum -y upgrade
#Updated: centos-release.x86_64 0:7-0.1406.el7.centos.2.6

systemctl stop firewalld.service;systemctl disable firewalld.service
getenforce
sed -i 's/enforcing/permissive/g' /etc/selinux/config
echo 0 > /sys/fs/selinux/enforce
```

192.168.X.184

juno-controller:/root/env.sh

```
CONTROLLER_IP=192.168.32.181
ADMIN_TOKEN=ADMIN
SERVICE_PWD=service
ADMIN_PWD=password
META_PWD=meta123
#juno-controller node
THISHOST_NAME=juno-controller
THISHOST_IP=192.168.32.181
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=na
THISHOST_TUNNEL_NETMASK=24
```

juno-network:/root/env.sh

```
CONTROLLER_IP=192.168.32.181
ADMIN_TOKEN=ADMIN
SERVICE_PWD=service
ADMIN_PWD=password
META_PWD=meta123
#juno-network node
THISHOST_NAME=juno-network
THISHOST_IP=192.168.32.182
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=192.168.33.
182
THISHOST_TUNNEL_NETMASK=24
```

juno-compute01/02:/root/env.sh

```
CONTROLLER_IP=192.168.32.181
ADMIN_TOKEN=ADMIN
SERVICE_PWD=service
ADMIN_PWD=password
META_PWD=meta123
#juno-compute node
THISHOST_NAME=juno-compute
THISHOST_IP=192.168.32.183
THISHOST_NETMASK=255.255.255.0
THISHOST_GATEWAY=192.168.32.1
THISHOST_DNS=192.168.32.1
THISHOST_TUNNEL_IP=192.168.33.
183
THISHOST_TUNNEL_NETMASK=24
```

Env setting

IP setting

- 먼저 각 node에서 env.sh를 실행한다. 그리고 아래대로 network-script를 실행한다.

All node

```
#get primary NIC info
for i in $(ls /sys/class/net); do
    NIC=$i
    MY_MAC=$(cat /sys/class/net/$i/address)
    if [ "$(cat /sys/class/net/$i/ifindex)" == '2' ]; then
        #setup the IP configuration for management NIC
        sed -i.bak "s/dhcp/none/g" /etc/sysconfig/network-scripts/ifcfg-$NIC
        sed -i "s/HWADDR/#HWADDR/g" /etc/sysconfig/network-scripts/ifcfg-$NIC
        sed -i "/#HWADDR/a HWADDR=\"$MY_MAC\"" /etc/sysconfig/network-scripts/ifcfg-$NIC
        sed -i "s/UUID/#UUID/g" /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "IPADDR=\"$THISHOST_IP\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "PREFIX=\"$THISHOST_NETMASK\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "GATEWAY=\"$THISHOST_GATEWAY\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "DNS1=\"$THISHOST_DNS\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        mv /etc/sysconfig/network-scripts/ifcfg-$NIC.bak .
    fi
    if [ "$(cat /sys/class/net/$i/ifindex)" == '3' ]; then
        #create config file for Tunnel NIC
        echo "HWADDR=\"$MY_MAC\"" > /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "TYPE=\"Ethernet\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "BOOTPROTO=\"none\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "IPV4_FAILURE_FATAL=\"no\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "NAME=\"$NIC\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "ONBOOT=\"yes\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "IPADDR=\"$THISHOST_TUNNEL_IP\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "PREFIX=\"$THISHOST_TUNNEL_NETMASK\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
    fi
    if [ "$(cat /sys/class/net/$i/ifindex)" == '4' ]; then
        #create config file for External NIC
        echo "HWADDR=\"$MY_MAC\"" > /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "TYPE=\"Ethernet\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "BOOTPROTO=\"none\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "IPV4_FAILURE_FATAL=\"no\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "NAME=\"$NIC\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
        echo "ONBOOT=\"yes\"" >> /etc/sysconfig/network-scripts/ifcfg-$NIC
    fi
done
```

```
#setup hostname
cp -f /dev/null /etc/hostname
echo " $THISHOST_NAME " >
/etc/hostname
echo "$THISHOST_IP
$THISHOST_NAME" >> /etc/hosts
```

CONTROLLER NODE SETTING

Openstack DB setting

- controller node = controller + MariaDB,RabbitMQ,Glance,NOVA api/scheduler,Neutron api, cinder api
- `egrep -v "^#|^$" /etc/my.cnf`

```
yum -y install mariadb mariadb-server MySQL-python
```

```
#edit /etc/my.cnf
```

```
sed -i.bak "10i\\  
bind-address = $CONTROLLER_IP\\n\\  
default-storage-engine =  
innodb\\n\\  
innodb_file_per_table\\n\\  
collation-server =  
utf8_general_ci\\n\\  
init-connect = 'SET NAMES  
utf8'\\n\\  
character-set-server = utf8\\n\\  
" /etc/my.cnf
```

```
[mysqld]  
bind-address = 192.168.32.181  
default-storage-engine = innodb  
innodb_file_per_table  
collation-server = utf8_general_ci  
init-connect = 'SET NAMES utf8'  
character-set-server = utf8
```

```
systemctl enable mariadb.service  
systemctl start mariadb.service  
mysql_secure_installation # mariadb 암호 설정  
Enter current password for root (enter for none): Enter  
OK, successfully used password, moving on...  
Setting the root password ensures that nobody can log into the MariaDB  
root user without the proper authorisation.  
Set root password? [Y/n] Y  
New password: password ; Re-enter new password: password  
Password updated successfully!  
Reloading privilege tables..  
... Success! # Enter *3
```

CONTROLLER NODE SETTING

Openstack DB 생성

database 생성 및 권한 설정

```
mysql -u root -p <<EOF
CREATE DATABASE nova;
CREATE DATABASE cinder;
CREATE DATABASE glance;
CREATE DATABASE keystone;
CREATE DATABASE neutron;
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY '$SERVICE_PWD';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' IDENTIFIED BY '$SERVICE_PWD';
FLUSH PRIVILEGES;
EOF
```

```
mysql -u root -p <<EOF
show databases;
EOF
```

Messaging server

```
yum -y install rabbitmq-server
systemctl enable rabbitmq-server.service
systemctl start rabbitmq-server.service
systemctl status rabbitmq-server.service
```

CONTROLLER NODE SETTING

keystone 설치

- keystone은 인증을 위한 서비스
- yum -y install openstack-keystone python-keystoneclient

```
#edit /etc/keystone.conf
```

```
sed -i.bak
"s/#admin_token=ADMIN/admin_token=$ADMIN_TOKEN/g"
/etc/keystone/keystone.conf

sed -i "\[database\]/a \
connection =
mysql://keystone:$SERVICE_PWD@$CONTROLLER_IP/keystone"
/etc/keystone/keystone.conf

sed -i "\[token\]/a \
provider = keystone.token.providers.uuid.Provider\n\
driver =
keystone.token.persistence.backends.sql.Token\n"
/etc/keystone/keystone.conf
sed -i "\[revoke\]/a \
driver =
keystone.contrib.revoke.backends.sql.Revoke\n"
/etc/keystone/keystone.conf
```

```
[DEFAULT]
# admin token 설정
admin_token=ADMIN
[database]
# database 접근 정보
connection =
mysql://keystone:service@192.168.32.181/keystone
[token]
#uuid token provider와 sql driver 정의
provider =
keystone.token.providers.uuid.Provider

driver =
keystone.token.persistence.backends.sql.Token
```

```
keystone-manage pki_setup --keystone-user keystone --keystone-group keystone # SSL PKI key 생성
```

```
chown -R keystone:keystone /var/log/keystone
chown -R keystone:keystone /etc/keystone/ssl
chmod -R o-rwx /etc/keystone/ssl
```

```
su -s /bin/sh -c "keystone-manage db_sync" keystone # 테이블 생성
```

```
#start keystone
systemctl enable openstack-keystone.service
systemctl start openstack-keystone.service
```

CONTROLLER NODE SETTING

create users and tenants

- Define users, tenants, and roles

```
export OS_SERVICE_TOKEN=$ADMIN_TOKEN
export OS_SERVICE_ENDPOINT=http://$CONTROLLER_IP:35357/v2.0
```

```
keystone tenant-create --name admin --description "Admin Tenant"
keystone user-create --name admin --pass $ADMIN_PWD
keystone role-create --name admin
keystone user-role-add --tenant admin --user admin --role admin
keystone role-create --name _member_
keystone user-role-add --tenant admin --user admin --role _member_
keystone tenant-create --name demo --description "Demo Tenant"
keystone user-create --name demo --pass password
keystone user-role-add --tenant demo --user demo --role _member_
keystone tenant-create --name service --description "Service Tenant"
keystone service-create --name keystone --type identity \
  --description "OpenStack Identity"
keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ identity / {print $2}') \
  --publicurl http://$CONTROLLER_IP:5000/v2.0 \
  --internalurl http://$CONTROLLER_IP:5000/v2.0 \
  --adminurl http://$CONTROLLER_IP:35357/v2.0 \
  --region regionOne #REST API 주소 생성

unset OS_SERVICE_TOKEN OS_SERVICE_ENDPOINT
```

CONTROLLER NODE SETTING

Authorized all node as admin

- 아래 셸을 모든 노드에 등록하여, admin이 항상 관리할 수 있도록 한다. 보통 .bashrc에 등록하여 사용함.

All node

```
echo "export OS_TENANT_NAME=admin" > keystone_admin
echo "export OS_USERNAME=admin" >> keystone_admin
echo "export OS_PASSWORD=$ADMIN_PWD" >> keystone_admin
echo "export OS_AUTH_URL=http://$CONTROLLER_IP:35357/v2.0">> keystone_admin
source keystone_admin
```

```
keystone user-list
keystone user-role-list
keystone tenant-list
keystone token-get
```


CONTROLLER NODE SETTING

glance-api / glance-registry / Database / Storage repository for image files

- Glance는 인스턴스의 운영체제 이미지 파일을 관리한다.
- yum -y install openstack-glance python-glanceclient

```
keystone user-create --name glance --pass $SERVICE_PWD
keystone user-role-add --user glance --tenant service --role admin
keystone service-create --name glance --type image \
  --description "OpenStack Image Service"
keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ image / {print $2}') \
  --publicurl http://$CONTROLLER_IP:9292 \
  --internalurl http://$CONTROLLER_IP:9292 \
  --adminurl http://$CONTROLLER_IP:9292 \
  --region regionOne
```

```
#edit /etc/glance/glance-api.conf
```

```
sed -i.bak "\[database\]/a \
connection =
mysql://glance:$SERVICE_PWD@$CONTROLLER_IP/glance"
/etc/glance/glance-api.conf
sed -i "\[keystone_auth\]/a \
auth_uri = http://$CONTROLLER_IP:5000/v2.0\n\
identity_uri = http://$CONTROLLER_IP:35357\n\
admin_tenant_name = service\n\
admin_user = glance\n\
admin_password = $SERVICE_PWD" /etc/glance/glance-
api.conf
sed -i "\[paste_deploy\]/a \
flavor = keystone" /etc/glance/glance-api.conf
sed -i "\[glance_store\]/a \
default_store = file\n\
filesystem_store_datadir = /var/lib/glance/images/"
/etc/glance/glance-api.conf
```

```
[DEFAULT]
[database]
connection =
mysql://glance:service@192.168.32.181/glance
[keystone_auth]
auth_uri = http://192.168.32.181:5000/v2.0
identity_uri = http://192.168.32.181:35357
admin_tenant_name = service
admin_user = glance
admin_password = service
[paste_deploy]
flavor = keystone
[glance_store]
default_store = file
filesystem_store_datadir=/var/lib/glance/im
ages/ # Cluster native mount point
```

CONTROLLER NODE SETTING

glance-api / glance-registry / Database / Storage repository for image files

- 이미지 등록을 담당하는 glance-registry 세팅 /

```
#edit /etc/glance/glance-registry.conf
```

```
sed -i.bak "\[database\]/a \  
connection =  
mysql://glance:$SERVICE_PWD@$CONTROLLER_IP/glance"  
/etc/glance/glance-registry.conf  
sed -i "\[keystone_authtoken\]/a \  
auth_uri = http://$CONTROLLER_IP:5000/v2.0\  
identity_uri = http://$CONTROLLER_IP:35357\  
admin_tenant_name = service\  
admin_user = glance\  
admin_password = $SERVICE_PWD" /etc/glance/glance-  
registry.conf  
sed -i "\[paste_deploy\]/a \  
flavor = keystone" /etc/glance/glance-registry.conf
```

```
[DEFAULT]  
[database]  
connection =  
mysql://glance:service@192.168.32.181/glance  
[keystone_authtoken]  
auth_uri = http://192.168.32.181:5000/v2.0  
identity_uri = http://192.168.32.181:35357  
admin_tenant_name = service  
admin_user = glance  
admin_password = service  
[paste_deploy]  
flavor = keystone  
[profiler]
```

```
#start glance  
su -s /bin/sh -c "glance-manage db_sync" glance  
systemctl enable openstack-glance-api.service openstack-glance-registry.service  
systemctl start openstack-glance-api.service openstack-glance-registry.service  
#upload the cirros image to glance  
yum -y install wget  
wget http://cdn.download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img  
glance image-create --name "cirros-0.3.3-x86_64" --file cirros-0.3.3-x86_64-disk.img \  
--disk-format qcow2 --container-format bare --is-public True --progress  
glance image-create --name 'centos7' --disk-format qcow2 --container-format bare --is-public true \  
--copy-from http://cloud.centos.org/centos/7/images/CentOS-7-x86_64-GenericCloud-20141129_01.qcow2
```

```
glance image-list
```

CONTROLLER NODE SETTING

nova-api / nova-compute / nova-scheduler / nova-conductor module

- 인스턴스 생성 및 삭제를 책임지는 nova 설치

```
keystone user-create --name nova --pass $SERVICE_PWD
keystone user-role-add --user nova --tenant service --role admin
keystone service-create --name nova --type compute \
  --description "OpenStack Compute"
keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ compute / {print $2}') \
  --publicurl http://$CONTROLLER_IP:8774/v2/%(tenant_id)s \
  --internalurl http://$CONTROLLER_IP:8774/v2/%(tenant_id)s \
  --adminurl http://$CONTROLLER_IP:8774/v2/%(tenant_id)s \
  --region regionOne
```

```
#install the nova controller components (To install and configure Compute controller components)
yum -y install openstack-nova-api openstack-nova-cert openstack-nova-conductor \
  openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler \
  python-novaclient
```

CONTROLLER NODE SETTING

nova-api / nova-compute / nova-scheduler / nova-conductor module

• Nova.conf 파일 구성

/ egrep -v "^#|^\$" /etc/nova/nova.conf

```
#edit /etc/nova/nova.conf
```

```
sed -i.bak "/[database\]/a \  
connection = mysql://nova:$SERVICE_PWD@$CONTROLLER_IP/nova"  
/etc/nova/nova.conf  
sed -i "/[DEFAULT\]/a \  
rpc_backend = rabbit\  
rabbit_host = $CONTROLLER_IP\  
auth_strategy = keystone\  
my_ip = $CONTROLLER_IP\  
vncserver_listen = $CONTROLLER_IP\  
vncserver_proxyclient_address = $CONTROLLER_IP\  
network_api_class = nova.network.neutronv2.api.API\  
security_group_api = neutron\  
linuxnet_interface_driver =  
nova.network.linux_net.LinuxOVSDriver\  
firewall_driver = nova.virt.firewall.NoopFirewallDriver"  
/etc/nova/nova.conf  
sed -i "/[keystone_auth\]/i \  
[database]connection = mysql://nova:$SERVICE_PWD@$CONTROLLER_IP/nova"  
/etc/nova/nova.conf  
sed -i "/[keystone_auth\]/a \  
auth_uri = http://$CONTROLLER_IP:5000/v2.0\  
identity_uri = http://$CONTROLLER_IP:35357\  
admin_tenant_name = service\  
admin_user = nova\  
admin_password = $SERVICE_PWD" /etc/nova/nova.conf  
sed -i "/[glance\]/a host = $CONTROLLER_IP" /etc/nova/nova.conf  
sed -i "/[neutron\]/a \  
url = http://$CONTROLLER_IP:9696\  
auth_strategy = keystone\  
admin_auth_url = http://$CONTROLLER_IP:35357/v2.0\  
admin_tenant_name = service\  
admin_username = neutron\  
admin_password = $SERVICE_PWD\  
service_metadata_proxy = True\  
metadata_proxy_shared_secret = $META_PWD" /etc/nova/nova.conf
```

```
[DEFAULT]  
rpc_backend = rabbit  
rabbit_host = 192.168.32.181  
auth_strategy = keystone  
my_ip = 192.168.32.181  
vncserver_listen = 192.168.32.181  
vncserver_proxyclient_address = 192.168.32.181  
network_api_class = nova.network.neutronv2.api.API  
security_group_api = neutron  
linuxnet_interface_driver =  
nova.network.linux_net.LinuxOVSDriver  
firewall_driver = nova.virt.firewall.NoopFirewallDriver  
[baremetal]  
[glance]  
host = 192.168.32.181  
[hyperv]  
[database]  
connection = mysql://nova:service@192.168.32.181/nova  
[keystone_auth]  
auth_uri = http://192.168.32.181:5000/v2.0  
identity_uri = http://192.168.32.181:35357  
admin_tenant_name = service  
admin_user = nova  
admin_password = service  
[neutron]  
url = http://192.168.32.181:9696  
auth_strategy = keystone  
admin_auth_url = http://192.168.32.181:35357/v2.0  
admin_tenant_name = service  
admin_username = neutron  
admin_password = service  
service_metadata_proxy = True  
metadata_proxy_shared_secret = meta123
```

CONTROLLER NODE SETTING

nova-api / nova-compute / nova-scheduler / nova-conductor module

● nova 서비스 구동

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

systemctl enable openstack-nova-api.service openstack-nova-cert.service \
openstack-nova-consoleauth.service openstack-nova-scheduler.service \
openstack-nova-conductor.service openstack-nova-novncproxy.service

systemctl start openstack-nova-api.service openstack-nova-cert.service \
openstack-nova-scheduler.service \
openstack-nova-conductor.service openstack-nova-novncproxy.service
```

openstack-neutron /openstack-neutron-ml2 / python-neutronclient

● Neutron 서버 설치

```
#create keystone entries for neutron
keystone user-create --name neutron --pass $SERVICE_PWD
keystone user-role-add --user neutron --tenant service --role admin
keystone service-create --name neutron --type network \
--description "OpenStack Networking"
keystone endpoint-create \
--service-id $(keystone service-list | awk '/ network / {print $2}') \
--publicurl http://$CONTROLLER_IP:9696 \
--internalurl http://$CONTROLLER_IP:9696 \
--adminurl http://$CONTROLLER_IP:9696 \
--region regionOne
```

```
#install neutron
yum -y install openstack-neutron openstack-neutron-ml2 python-neutronclient
```

CONTROLLER NODE SETTING

openstack-neutron /openstack-neutron-ml2 / python-neutronclient

● Neutron 서버를 설치함

```
/ egrep -v "^#|^$" /etc/neutron/neutron.conf
```

```
#edit /etc/neutron/neutron.conf
```

```
sed -i.bak "/\[database\]/a \  
connection =  
mysql://neutron:$SERVICE_PWD@$CONTROLLER_IP/neutron"  
/etc/neutron/neutron.conf  
  
SERVICE_TENANT_ID=$(keystone tenant-list | awk '/ service  
/ {print $2}')  
sed -i '0,/\[DEFAULT\]/s//\[DEFAULT\]\  
rpc_backend = rabbit\  
rabbit_host = "$CONTROLLER_IP"  
auth_strategy = keystone\  
core_plugin = ml2\  
service_plugins = router\  
allow_overlapping_ips = True\  
notify_nova_on_port_status_changes = True\  
notify_nova_on_port_data_changes = True\  
nova_url = http://\'"$CONTROLLER_IP"':8774/v2\  
nova_admin_auth_url =  
http://\'"$CONTROLLER_IP"':35357/v2.0\  
nova_region_name = regionOne\  
nova_admin_username = nova\  
nova_admin_tenant_id = "$SERVICE_TENANT_ID"  
nova_admin_password = "$SERVICE_PWD"/'  
/etc/neutron/neutron.conf  
  
sed -i "/\[keystone_authtoken\]/a \  
auth_uri = http://$CONTROLLER_IP:5000/v2.0\  
identity_uri = http://$CONTROLLER_IP:35357\  
admin_tenant_name = service\  
admin_user = neutron\  
admin_password = $SERVICE_PWD" /etc/neutron/neutron.conf
```

```
[DEFAULT]  
rpc_backend = rabbit  
rabbit_host = 192.168.32.181  
auth_strategy = keystone  
core_plugin = ml2  
service_plugins = router  
allow_overlapping_ips = True  
notify_nova_on_port_status_changes = True  
notify_nova_on_port_data_changes = True  
nova_url = http://192.168.32.181:8774/v2  
nova_admin_auth_url = http://192.168.32.181:35357/v2.0  
nova_region_name = regionOne  
nova_admin_username = nova  
nova_admin_tenant_id = 2ec220d040994c4589fb60afc98fc5c3  
nova_admin_password = service  
[matchmaker_redis]  
[matchmaker_ring]  
[quotas]  
[agent]  
[keystone_authtoken]  
auth_uri = http://192.168.32.181:5000/v2.0  
identity_uri = http://192.168.32.181:35357  
admin_tenant_name = service  
admin_user = neutron  
admin_password = service  
[database]  
connection = mysql://neutron:service@192.168.32.181/neutron  
connection = mysql://neutron:service@192.168.32.181/neutron  
connection = mysql://neutron:service@192.168.32.181/neutron
```

CONTROLLER NODE SETTING

openstack-neutron /openstack-neutron-ml2 / python-neutronclient

- Neutron 기본 plug-in인 ML2 사용 / GRE/Openvswitch 사용
- / egrep -v "^#|^\$" /etc/neutron/plugins/ml2/ml2_conf.ini

```
#edit /etc/neutron/plugins/ml2/ml2_conf.ini
```

```
#edit /etc/neutron/plugins/ml2/ml2_conf.ini
sed -i "/\[ml2\]/a \
type_drivers = flat,gre\n\
tenant_network_types = gre\n\
mechanism_drivers = openvswitch"
/etc/neutron/plugins/ml2/ml2_conf.ini

sed -i "/\[ml2_type_gre\]/a \
tunnel_id_ranges = 1:1000"
/etc/neutron/plugins/ml2/ml2_conf.ini

sed -i "/\[securitygroup\]/a \
enable_security_group = True\n\
enable_ipset = True\n\
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFire
ewallDriver" /etc/neutron/plugins/ml2/ml2_conf.ini
```

```
[ml2]
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = openvswitch
[ml2_type_gre]
tunnel_id_ranges = 1:1000
[securitygroup]
enable_security_group = True
enable_ipset = True
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFire
wallDriver
```

```
#start neutron
ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf \
--config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade jun0" neutron
systemctl restart openstack-nova-api.service openstack-nova-scheduler.service \
openstack-nova-conductor.service
systemctl enable neutron-server.service
systemctl start neutron-server.service
```

CONTROLLER NODE SETTING

• `egrep -v "^#|^$" /etc/openstack-dashboard/local_settings`

openstack-dashboard httpd mod_wsgi memcached python-memcached

• 웹기반 인터페이스인 dashboard 설치

```
yum -y install openstack-dashboard httpd mod_wsgi memcached python-memcached
```

```
#edit /etc/openstack-dashboard/local_settings
```

```
sed -i.bak "s/ALLOWED_HOSTS = \['horizon.example.com',  
'localhost'\]/ALLOWED_HOSTS = ['*']/" /etc/openstack-  
dashboard/local_settings  
sed -i 's/OPENSTACK_HOST = "127.0.0.1"/OPENSTACK_HOST =  
"'"$CONTROLLER_IP"'"/' /etc/openstack-  
dashboard/local_settings
```

```
ALLOWED_HOSTS = ['*']  
OPENSTACK_HOST = "192.168.32.181"
```

```
#start dashboard  
setsebool -P httpd_can_network_connect on  
chown -R apache:apache /usr/share/openstack-dashboard/static  
systemctl enable httpd.service memcached.service  
systemctl start httpd.service memcached.service
```


CONTROLLER NODE SETTING

openstack-cinder/ python-cinderclient / python-oslo-db

- 인스턴스 블록스토리를 관제를 cinder controller 설치

```
#create keystone entries for cinder
keystone user-create --name cinder --pass $SERVICE_PWD
keystone user-role-add --user cinder --tenant service --role admin
keystone service-create --name cinder --type volume \
  --description "OpenStack Block Storage"

keystone service-create --namecinderv2 --type volumev2 \
  --description "OpenStack Block Storage"

keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ volume / {print $2}') \
  --publicurl http://$CONTROLLER_IP:8776/v1/%\(tenant_id\)s \
  --internalurl http://$CONTROLLER_IP:8776/v1/%\(tenant_id\)s \
  --adminurl http://$CONTROLLER_IP:8776/v1/%\(tenant_id\)s \
  --region regionOne

keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ volumev2 / {print $2}') \
  --publicurl http://$CONTROLLER_IP:8776/v2/%\(tenant_id\)s \
  --internalurl http://$CONTROLLER_IP:8776/v2/%\(tenant_id\)s \
  --adminurl http://$CONTROLLER_IP:8776/v2/%\(tenant_id\)s \
  --region regionOne

#install cinder controller
yum -y install openstack-cinder python-cinderclient python-oslo-db
```

CONTROLLER NODE SETTING

openstack-cinder/ python-cinderclient / python-oslo-db

cinder controller 설치

/ egrep -v "^#|^\$" /etc/cinder/cinder.conf

```
#edit /etc/cinder/cinder.conf
```

```
sed -i.bak "/\[database\]/a connection =
mysql://cinder:$SERVICE_PWD@$CONTROLLER_IP/cinder" /etc/cinder/cinder.conf
```

```
sed -i "0,/\[DEFAULT\]/a \
rpc_backend = rabbit\n\
rabbit_host = $CONTROLLER_IP\n\
auth_strategy = keystone\n\
my_ip = $CONTROLLER_IP" /etc/cinder/cinder.conf
```

```
sed -i "/\[keystone_authtoken\]/a \
auth_uri = http://$CONTROLLER_IP:5000/v2.0\n\
identity_uri = http://$CONTROLLER_IP:35357\n\
admin_tenant_name = service\n\
admin_user = cinder\n\
admin_password = $SERVICE_PWD"
/etc/cinder/cinder.conf
```

[**DEFAULT**]

```
rpc_backend = rabbit
rabbit_host = 192.168.32.181
auth_strategy = keystone
my_ip = 192.168.32.181
```

[**database**]

```
connection = mysql://cinder:service@192.168.32.181/cinder
```

[**keystone_authtoken**]

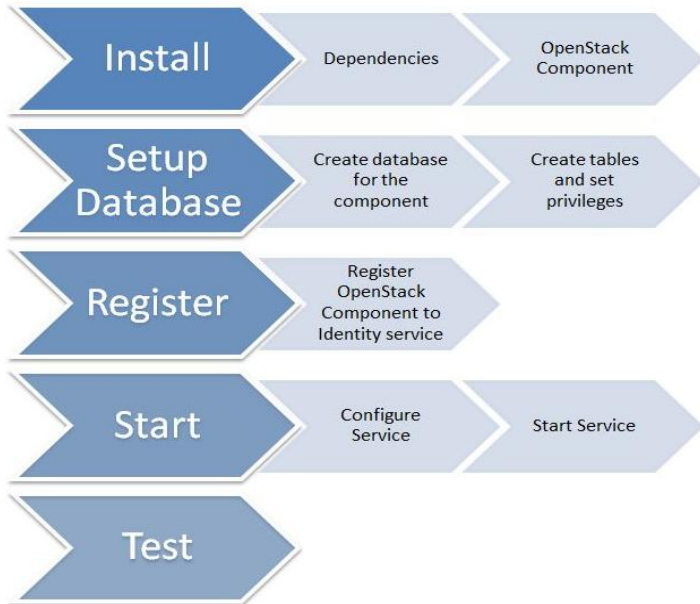
```
auth_uri = http://192.168.32.181:5000/v2.0
identity_uri = http://192.168.32.181:35357
admin_tenant_name = service
admin_user = cinder
admin_password = service
```

CONTROLLER NODE SETTING

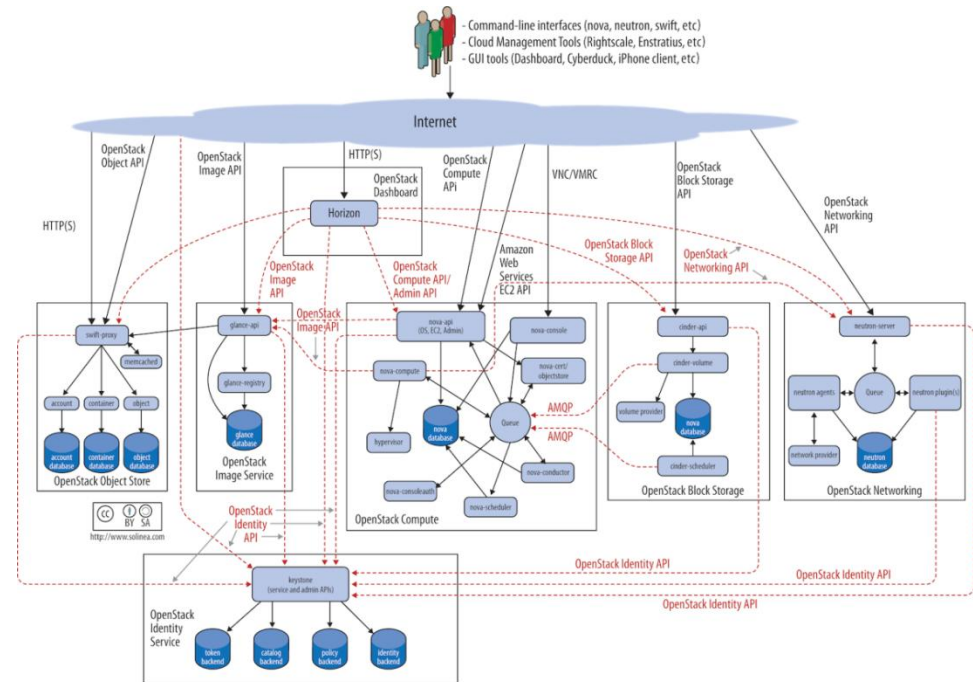
openstack-cinder/ python-cinderclient / python-oslo-db

cinder controller start

```
#start cinder controller  
su -s /bin/sh -c "cinder-manage db sync" cinder  
systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service  
systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service
```



OpenStack Icehouse Installation and Configuration Flow



NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

```
echo 'net.ipv4.ip_forward=1' >> /etc/sysctl.conf
echo 'net.ipv4.conf.all.rp_filter=0' >> /etc/sysctl.conf
echo 'net.ipv4.conf.default.rp_filter=0' >> /etc/sysctl.conf
sysctl -p
```

NETWORK NODE

openstack-neutron neutron-ml2 neutron-openswitch

- Controller에 openswitch와 L3 부분이 추가된다고 생각하면 된다 . 다른 부분은 진한 글씨체임
- `egrep -v "^#|^$" /etc/neutron/neutron.conf / egrep -v "^#|^$" /etc/neutron/l3_agent.ini`

```
yum -y install openstack-neutron openstack-neutron-ml2 openstack-neutron-openswitch
```

```
#edit /etc/neutron/neutron.conf
```

```
sed -i '0,/\[DEFAULT\]/s/\[DEFAULT\]\
rpc_backend = rabbit\
rabbit_host = "$CONTROLLER_IP"\
auth_strategy = keystone\
core_plugin = ml2\
service_plugins = router\
allow_overlapping_ips = True/'
/etc/neutron/neutron.conf
sed -i "/\[keystone_authtoken\]/a \
auth_uri = http://$CONTROLLER_IP:5000/v2.0\n\
identity_uri = http://$CONTROLLER_IP:35357\n\
admin_tenant_name = service\n\
admin_user = neutron\n\
admin_password = $SERVICE_PWD"
/etc/neutron/neutron.conf
```

```
[DEFAULT]
rpc_backend = rabbit
rabbit_host = 192.168.0.181
auth_strategy = keystone
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True

[matchmaker_redis]
```

```
[keystone_authtoken]
auth_uri = http://192.168.0.181:5000/v2.0
identity_uri = http://192.168.0.181:35357
admin_tenant_name = service
admin_user = neutron
admin_password = service
```

```
#edit /etc/neutron/l3_agent.ini
```

```
sed -i "/\[DEFAULT\]/a \
interface_driver =
neutron.agent.linux.interface.OVSInterfaceDriver\n\
use_namespaces = True\n\
external_network_bridge = br-ex"
/etc/neutron/l3_agent.ini
```

```
[DEFAULT]
interface_driver =
neutron.agent.linux.interface.OVSInterfaceDriver
use_namespaces = True
external_network_bridge = br-ex
```

NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

Openvswitch setting

/ egrep -v "^#|^\$" /etc/neutron/plugins/ml2/ml2_conf.ini

```
#edit /etc/neutron/plugins/ml2/ml2_conf.ini
```

```
sed -i "\[ml2\]/a \  
type_drivers = flat,gre\  
tenant_network_types = gre\  
mechanism_drivers = openvswitch"  
/etc/neutron/plugins/ml2/ml2_conf.ini  
sed -i "\[ml2_type_flat\]/a \  
flat_networks = external"  
/etc/neutron/plugins/ml2/ml2_conf.ini  
sed -i "\[ml2_type_gre\]/a \  
tunnel_id_ranges = 1:1000"  
/etc/neutron/plugins/ml2/ml2_conf.ini  
sed -i "\[securitygroup\]/a \  
enable_security_group = True\  
enable_ipset = True\  
firewall_driver =  
neutron.agent.linux.iptables_firewall.OVSHybridI  
ptablesFirewallDriver\  
[ovs]\  
local_ip = $THISHOST_TUNNEL_IP\  
enable_tunneling = True\  
bridge_mappings = external:br-ex\  
[agent]\  
tunnel_types = gre"  
/etc/neutron/plugins/ml2/ml2_conf.ini
```

```
[ml2]  
type_drivers = flat,gre  
tenant_network_types = gre  
mechanism_drivers = openvswitch  
[ml2_type_flat]  
flat_networks = external  
[ml2_type_vlan]  
[ml2_type_gre]  
tunnel_id_ranges = 1:1000  
[ml2_type_vxlan]  
[securitygroup]  
enable_security_group = True  
enable_ipset = True  
firewall_driver =  
neutron.agent.linux.iptables_firewall.OVSHybridIp  
tablesFirewallDriver  
[ovs]  
local_ip = 192.168.33.182  
enable_tunneling = True  
bridge_mappings = external:br-ex  
[agent]  
tunnel_types = gre
```

NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

Openvswitch setting

```
egrep -v "^#|^$" / etc/neutron/dhcp_agent.ini
```

```
#edit /etc/neutron/dhcp_agent.ini
```

```
sed -i "/\[DEFAULT\]/a \  
interface_driver =  
neutron.agent.linux.interface.OVSInterfaceDriver  
\n\  
use_namespaces = True\n\  
external_network_bridge = br-ex"  
/etc/neutron/l3_agent.ini
```

```
sed -i "/\[DEFAULT\]/a \  
interface_driver =  
neutron.agent.linux.interface.OVSInterfaceDriver  
\n\  
dhcp_driver =  
neutron.agent.linux.dhcp.Dnsmasq\n\  
use_namespaces = True"  
/etc/neutron/dhcp_agent.ini
```

```
[root@juno-network neutron]# egrep -v  
"^#|^$" /etc/neutron/l3_agent.ini  
[DEFAULT]  
interface_driver =  
neutron.agent.linux.interface.OVSInterface  
Driver  
use_namespaces = True  
external_network_bridge = br-ex  
interface_driver =  
neutron.agent.linux.interface.OVSInterface  
Driver  
use_namespaces = True  
external_network_bridge = br-ex
```

NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

metadata_agent 정보 입력

```
#edit /etc/neutron/metadata_agent.ini
```

```
sed -i "s/auth_url/#auth_url/g"
/etc/neutron/metadata_agent.ini
sed -i "s/auth_region/#auth_region/g"
/etc/neutron/metadata_agent.ini
sed -i
"s/admin_tenant_name/#admin_tenant_name/g"
/etc/neutron/metadata_agent.ini
sed -i "s/admin_user/#admin_user/g"
/etc/neutron/metadata_agent.ini
sed -i "s/admin_password/#admin_password/g"
/etc/neutron/metadata_agent.ini
sed -i "/\[DEFAULT\]/a \
auth_url = http://$CONTROLLER_IP:5000/v2.0\n\
auth_region = regionOne\n\
admin_tenant_name = service\n\
admin_user = neutron\n\
admin_password = $SERVICE_PWD\n\
nova_metadata_ip = $CONTROLLER_IP\n\
metadata_proxy_shared_secret = $META_PWD"
/etc/neutron/metadata_agent.ini
```

```
[DEFAULT]
auth_url = http://192.168.32.181:5000/v2.0
auth_region = regionOne
admin_tenant_name = service
admin_user = neutron
admin_password = service
nova_metadata_ip = 192.168.32.181
metadata_proxy_shared_secret = meta123
```


NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

- NIC 외부네트워크 연결

```
#get external NIC info
for i in $(ls /sys/class/net); do
    if [ "$(cat /sys/class/net/$i/ifindex)" == '4' ]; then
        NIC=$i
        MY_MAC=$(cat /sys/class/net/$i/address)
        echo "$i ($MY_MAC)"
    fi
done
systemctl enable openvswitch.service
systemctl start openvswitch.service
ovs-vsctl add-br br-ex
ovs-vsctl add-port br-ex $NIC
ethtool -K $NIC gro off
```

NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

외부 network setting

```
[root@net01 network-scripts]# cat ifcfg-br-ex
TYPE=OVSIIntPort
OVS_BRIDGE=br-ex
DEVICETYPE=ovs
BOOTPROTO=none
IPADDR0=192.168.0.182
PREFIX0=24
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=no
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
NAME=br-ex
#UUID=33d13b63-9eba-4414-996a-75391a71fc6a
DEVICE=br-ex
ONBOOT=yes
GATEWAY=192.168.0.1
DNS1=8.8.8.8
```

```
[root@net01 network-scripts]# cat ifcfg-eth2
#HWADDR=00:19:99:D5:AA:D0
TYPE=OVSPort
DEVICETYPE=ovs
OVS_BRIDGE=br-ex
BOOTPROTO=none
NAME=eth2
#UUID=33d13b63-9eba-4414-996a-75391a71fc6a
DEVICE=eth2
ONBOOT=yes
```

```
ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
cp /usr/lib/systemd/system/neutron-openvswitch-agent.service \
  /usr/lib/systemd/system/neutron-openvswitch-agent.service.orig
sed -i 's,plugins/openvswitch/ovs_neutron_plugin.ini,plugin.ini,g' \
  /usr/lib/systemd/system/neutron-openvswitch-agent.service
```

NETWORK NODE

openstack-neutron neutron-ml2 neutron-openvswitch

- 서비스 활성화

```
systemctl enable neutron-openvswitch-agent.service neutron-l3-agent.service \  
neutron-dhcp-agent.service neutron-metadata-agent.service \  
neutron-ovs-cleanup.service  
systemctl start neutron-openvswitch-agent.service neutron-l3-agent.service \  
neutron-dhcp-agent.service neutron-metadata-agent.service
```

COMPUTE NODE

`nova-compute sysfsutils libvirt-daemon-config-nwfilter`

- Compute 클라우드 컴퓨팅을 위한 nova 설치

```
echo 0 > /sys/fs/selinux/enforce
echo 'net.ipv4.conf.all.rp_filter=0' >> /etc/sysctl.conf
echo 'net.ipv4.conf.default.rp_filter=0' >> /etc/sysctl.conf
sysctl -p
```

```
yum -y install openstack-nova-compute sysfsutils libvirt-daemon-config-nwfilter
```

COMPUTE NODE

nova-compute sysfsutils libvirt-daemon-config-nwfilter

/etc/nova/nova.conf

/ egrep -v "^#|^\$" /etc/nova/nova.conf

```
sed -i.bak "/\[DEFAULT\])/a \  
rpc_backend = rabbit\  
rabbit_host = $CONTROLLER_IP\  
auth_strategy = keystone\  
my_ip = $THISHOST_IP\  
vnc_enabled = True\  
vncserver_listen = 0.0.0.0\  
vncserver_proxyclient_address = $THISHOST_IP\  
novncproxy_base_url =  
http://$CONTROLLER_IP:6080/vnc_auto.html\  
network_api_class =  
nova.network.neutronv2.api.API\  
security_group_api = neutron\  
linuxnet_interface_driver =  
nova.network.linux_net.LinuxOVSInterfaceDriver\  
\  
firewall_driver =  
nova.virt.firewall.NoopFirewallDriver"  
/etc/nova/nova.conf  
sed -i "/\[keystone_auth_token\])/a \  
auth_uri = http://$CONTROLLER_IP:5000/v2.0\  
identity_uri = http://$CONTROLLER_IP:35357\  
admin_tenant_name = service\  
admin_user = nova\  
admin_password = $SERVICE_PWD"  
/etc/nova/nova.conf  
sed -i "/\[glance\])/a host = $CONTROLLER_IP"  
/etc/nova/nova.conf
```

```
[DEFAULT]  
rpc_backend = rabbit  
rabbit_host = 192.168.32.181  
auth_strategy = keystone  
my_ip = 192.168.32.183  
vnc_enabled = True  
vncserver_listen = 0.0.0.0  
vncserver_proxyclient_address = 192.168.32.183  
novncproxy_base_url =  
http://192.168.32.181:6080/vnc_auto.html  
network_api_class = nova.network.neutronv2.api.API  
security_group_api = neutron  
linuxnet_interface_driver =  
nova.network.linux_net.LinuxOVSInterfaceDriver  
firewall_driver =  
nova.virt.firewall.NoopFirewallDriver  
[glance]  
host = 192.168.32.181  
[keystone_auth_token]  
auth_uri = http://192.168.32.181:5000/v2.0  
identity_uri = http://192.168.32.181:35357  
admin_tenant_name = service  
admin_user = nova  
admin_password = service  
[neutron]  
url = http://192.168.32.181:9696  
auth_strategy = keystone  
admin_auth_url = http://192.168.32.181:35357/v2.0  
admin_tenant_name = service  
admin_username = neutron  
admin_password = service
```

COMPUTE NODE

openstack-neutron-m12 openstack-neutron-openvswitch

neutron setting

/ egrep -v "^#|^\$" /etc/neutron/neutron.conf

```
#install neutron
yum -y install openstack-neutron-m12 openstack-neutron-openvswitch
```

```
#edit /etc/neutron/neutron.conf
```

```
sed -i '0,/\[DEFAULT\]/s//\[DEFAULT\]\
rpc_backend = rabbit\n\
rabbit_host = "$CONTROLLER_IP"\n\
auth_strategy = keystone\n\
core_plugin = ml2\n\
service_plugins = router\n\
allow_overlapping_ips = True/'
/etc/neutron/neutron.conf
sed -i "/\[keystone_authtoken\]/a \
auth_uri = http://$CONTROLLER_IP:5000/v2.0\n\
identity_uri = http://$CONTROLLER_IP:35357\n\
admin_tenant_name = service\n\
admin_user = neutron\n\
admin_password = $SERVICE_PWD"
/etc/neutron/neutron.conf
```

```
[DEFAULT]
rpc_backend = rabbit
rabbit_host = 192.168.32.181
auth_strategy = keystone
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
```

```
[keystone_authtoken]
auth_uri = http://192.168.32.181:5000/v2.0
identity_uri = http://192.168.32.181:35357
admin_tenant_name = service
admin_user = neutron
admin_password = service
```

COMPUTE NODE

openstack-neutron-ml2 openstack-neutron-openvswitch

ml2_conf.ini 수정

/ egrep -v "^#|^\$" /etc/neutron/plugins/ml2/ml2_conf.ini

```
#edit /etc/neutron/plugins/ml2/ml2_conf.ini
```

```
sed -i "/\[ml2\]/a \  
type_drivers = flat,gre\  
tenant_network_types = gre\  
mechanism_drivers = openvswitch"  
/etc/neutron/plugins/ml2/ml2_conf.ini  
sed -i "/\[ml2_type_gre\]/a \  
tunnel_id_ranges = 1:1000"  
/etc/neutron/plugins/ml2/ml2_conf.ini  
sed -i "/\[securitygroup\]/a \  
enable_security_group = True\  
enable_ipset = True\  
firewall_driver =  
neutron.agent.linux.iptables_firewall.OVSHybridI  
ptablesFirewallDriver\  
[ovs]\  
local_ip = $THISHOST_TUNNEL_IP\  
enable_tunneling = True\  
[agent]\  
tunnel_types = gre"  
/etc/neutron/plugins/ml2/ml2_conf.ini
```

```
[ml2]  
type_drivers = flat,gre  
tenant_network_types = gre  
mechanism_drivers = openvswitch  
[ml2_type_gre]  
tunnel_id_ranges = 1:1000  
[securitygroup]  
enable_security_group = True  
enable_ipset = True  
firewall_driver =  
neutron.agent.linux.iptables_firewall.OVSHybridIp  
tablesFirewallDriver  
[ovs]  
local_ip = 192.168.33.183  
enable_tunneling = True  
[agent]  
tunnel_types = gre
```

```
systemctl enable openvswitch.service  
systemctl start openvswitch.service
```

COMPUTE NODE

nova-compute node 구성

- /etc/nova/nova.conf

```
#edit /etc/nova/nova.conf
```

```
sed -i "/\[neutron\]/a \  
url = http://$CONTROLLER_IP:9696\  
auth_strategy = keystone\  
admin_auth_url =  
http://$CONTROLLER_IP:35357/v2.0\  
admin_tenant_name = service\  
admin_username = neutron\  
admin_password = $SERVICE_PWD"  
/etc/nova/nova.conf
```

```
[neutron]  
url = http://192.168.32.181:9696  
auth_strategy = keystone  
admin_auth_url = http://192.168.32.181:35357/v2.0  
admin_tenant_name = service  
admin_username = neutron  
admin_password = service
```

```
ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini  
cp /usr/lib/systemd/system/neutron-openvswitch-agent.service \  
/usr/lib/systemd/system/neutron-openvswitch-agent.service.orig  
sed -i 's,plugins/openvswitch/ovs_neutron_plugin.ini,plugin.ini,g' \  
/usr/lib/systemd/system/neutron-openvswitch-agent.service  
  
systemctl enable libvirtd.service openstack-nova-compute.service  
systemctl start libvirtd.service  
systemctl start openstack-nova-compute.service  
systemctl enable neutron-openvswitch-agent.service  
systemctl start neutron-openvswitch-agent.service
```


COMPUTE NODE

openstack-cinder targetcli python-oslo-db MySQL-python

● Cinder 디스크추가

```
#cinder storage node
```

```
pvcreate /dev/sdb
```

```
vgcreate cinder-volumes /dev/sdb
```

```
yum -y install openstack-cinder targetcli python-oslo-db MySQL-python
```

```
#edit /etc/cinder/cinder.conf
```

```
sed -i.bak "/\[database\]/a connection =
mysql://cinder:$SERVICE_PWD@$CONTROLLER_IP/cinder" /etc/cinder/cinder.conf
sed -i '0,/\[DEFAULT\]/s//\[DEFAULT\]\
rpc_backend = rabbit\
rabbit_host = "$CONTROLLER_IP"\
auth_strategy = keystone\
my_ip = "$THISHOST_IP"\
iscsi_helper = lioadm/' /etc/cinder/cinder.conf
sed -i "/\[keystone_authtoken\]/a \
auth_uri = http://$CONTROLLER_IP:5000/v2.0\n\
identity_uri = http://$CONTROLLER_IP:35357\n\
admin_tenant_name = service\n\
admin_user = cinder\n\
admin_password = $SERVICE_PWD"
/etc/cinder/cinder.conf
```

```
[DEFAULT]
```

```
rpc_backend = rabbit
rabbit_host = 192.168.32.181
auth_strategy = keystone
my_ip = 192.168.32.183
iscsi_helper = lioadm
```

```
[database]
```

```
connection =
mysql://cinder:service@192.168.32.181/cinder
[keystone_authtoken]
auth_uri = http://192.168.32.181:5000/v2.0
identity_uri = http://192.168.32.181:35357
admin_tenant_name = service
admin_user = cinder
admin_password = service
```

COMPUTE NODE

openstack-cinder targetcli python-oslo-db MySQL-python

Cinder 디스크추가

```
sed -i 's/filter/#filter/g' /etc/lvm/lvm.conf
sed -i "/devices {/a \
filter = [\"a/sd/\", \"a/sdb/\", \"r/.*/\"] \"
/etc/lvm/lvm.conf
```

```
[root@juno-controller lvm]# grep filter
lvm.conf
filter = [ \"a/sda/\", \"a/sdb/\", \"r/.*/\"]
```

```
cinder create --display_name test3 2
```

```
[root@juno-compute cinder]# cinder list
```

ID	Status	Display Name	Size	Volume Type	Bootable	Attached to
35e69e09-015b-472e-a77c-a06f307beb92	available	test3	2	None	false	

```
[root@juno-compute cinder]# vgs
```

VG	#PV	#LV	#SN	Attr	VSize	VFree
centos	1	2	0	wz--n-	19.51g	0
cinder-volumes	1	1	0	wz--n-	50.00g	48.00g

```
[root@juno-compute cinder]# lvs
```

LV	VG	Attr	LSize	Pool	Origin	Data%	Move	Log	Cpy%	Sync
Convert										
root	centos	-wi-ao----	17.51g							
swap	centos	-wi-ao----	2.00g							
volume-35e69e09-015b-472e-a77c-a06f307beb92	cinder-volumes	-wi-a-----	2.00g							

```
systemctl enable openstack-cinder-volume.service target.service
systemctl start openstack-cinder-volume.service target.service
```

Contents

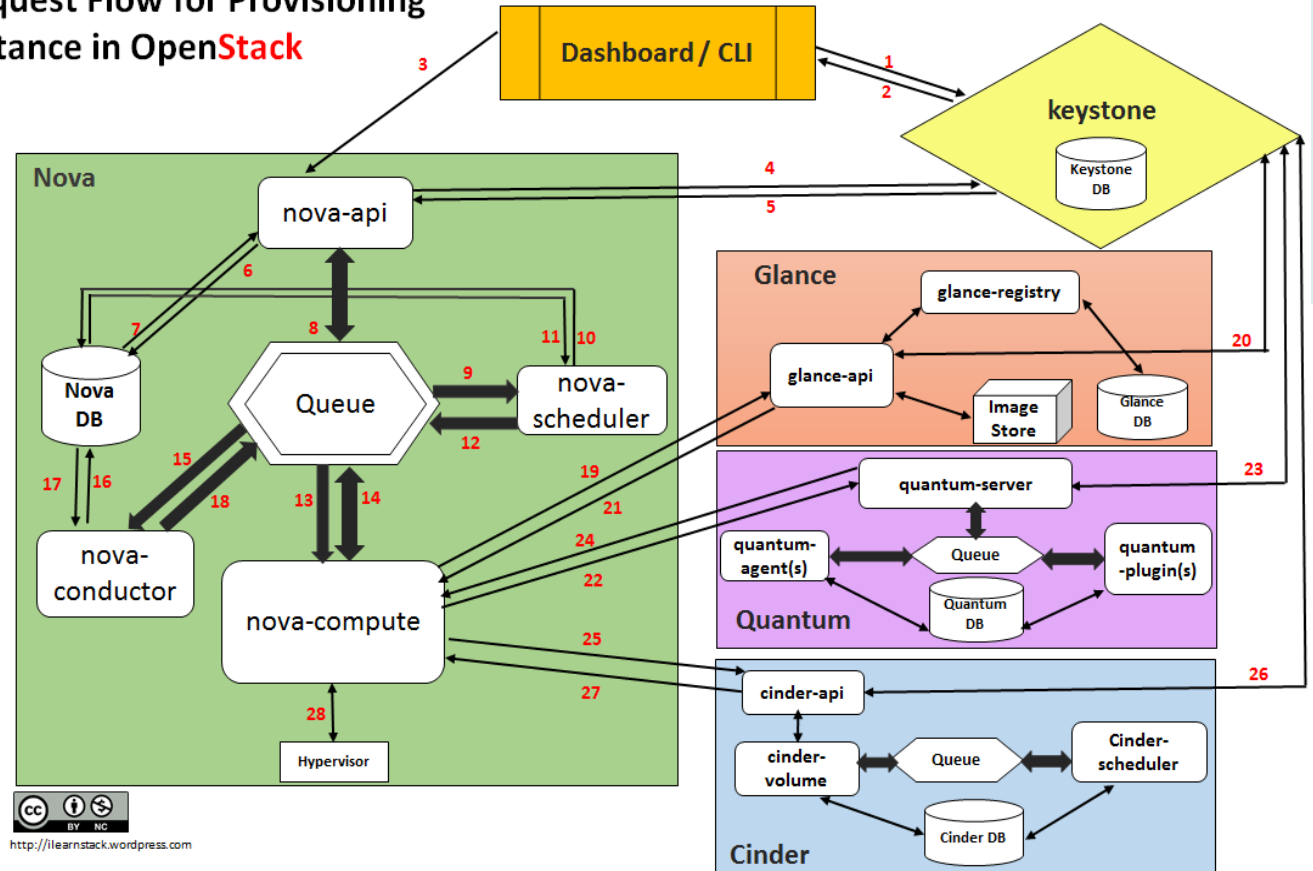
1. Openstack 인프라 구축 (4 node 구성) [30분]
2. Openstack 위에 VM 생성 [20분]
3. docker 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. Docker로 WEB서비스 구축 [15분]
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. Docker로 jenkins 구현 [30분]

What to do during 20 min.

OpenStack vm 생성

- 기본적인 vm 생성 방법
 - neutron network 생성
 - 라우터 생성
 - 게이트웨이 설정
 - 내부 인터페이스 추가
 - 인스턴스 생성

Request Flow for Provisioning Instance in OpenStack



Network만들기

Network 생성 / 외부 subnet 생성

- neutron net-create ext-net --shared --router:external True

```
[root@juno-controller ~]# Created a new network:
```

Field	Value
admin_state_up	True
id	74cea9a5-434c-4bff-89b7-a1e503b43d39
name	ext-net
provider:network_type	gre
provider:physical_network	
provider:segmentation_id	2
router:external	True
shared	True
status	ACTIVE
subnets	
tenant_id	e7cb7856091d4d839031d79582c93a76

네트워크 생성

이름: ext_net

설명: (empty)

프로젝트: (empty)

프로젝트 선택: (empty)

Provider 네트워크 타입: GRE

구분 ID: 2

관리자 상태: UP

공유:

외부 네트워크:

뒤로 네트워크 생성

- neutron subnet-create ext-net --name ext-subnet --allocation-pool start=192.168.0.200,end=192.168.0.220 --disable-dhcp --gateway 192.168.0.1 192.168.0.0/24

```
Created a new subnet:
```

Field	Value
allocation_pools	{ "start": "192.168.0.200", "end": "192.168.0.220" }
cidr	192.168.0.0/24
dns_nameservers	
enable_dhcp	False
gateway_ip	192.168.0.1
host_routes	
id	d84f7826-ae27-420f-9f1d-da7261c76e0f
ip_version	4
ipv6_address_mode	
ipv6_ra_mode	
name	ext-subnet
network_id	74cea9a5-434c-4bff-89b7-a1e503b43d39
tenant_id	e7cb7856091d4d839031d79582c93a76

서브넷 생성

서브넷 이름: ext-subnet

네트워크 주소: 192.168.0.0/24

IP 버전: IPv4

게이트웨이 IP: 192.168.0.1

게이트웨이 비활성:

서브넷 생성

Allocation Pools: 192.168.0.200-192.168.0.220

OS 대입 시제: (empty)

호스트 라우터: (empty)

뒤로 생성

Network만들기

내부 network 생성 / 내부 subnet 생성

neutron net-create admin-net

Created a new network:

Field	Value
admin_state_up	True
id	666c4f98-2a42-46a0-838c-0a82f7335585
name	admin-net
provider:network_type	gre
provider:physical_network	
provider:segmentation_id	1
router:external	False
shared	False
status	ACTIVE
subnets	
tenant_id	e7cb7856091d4d839031d79582c93a76

neutron subnet-create admin-net --name admin-subnet --gateway 10.0.1.1 10.0.1.0/24

Created a new subnet:

Field	Value
allocation_pools	{"start": "10.0.1.2", "end": "10.0.1.254"}
cidr	10.0.1.0/24
dns_nameservers	
enable_dhcp	True
gateway_ip	10.0.1.1
host_routes	
id	768d888e-9fee-46ac-9c98-bf2ba82d8a44
ip_version	4
ipv6_address_mode	
ipv6_ra_mode	
name	admin-subnet
network_id	666c4f98-2a42-46a0-838c-0a82f7335585
tenant_id	e7cb7856091d4d839031d79582c93a76

Network만들기

라우터 생성 / 외부라우터 연결

neutron router-create admin-router

Created a new router:

Field	Value
admin_state_up	True
distributed	False
external_gateway_info	
ha	False
id	14e32baa-bab5-4f72-9f19-ebd00f5d3671
name	admin-router
routes	
status	ACTIVE
tenant_id	e7cb7856091d4d839031d79582c93a76

라우터 생성

라우터 이름 *

취소 라우터 생성

neutron router-interface-add admin-router admin-subnet

Added interface b5913593-5e66-44cb-8d4a-88ae7c803162 to router admin-router. II

인터페이스 추가

서브넷 *

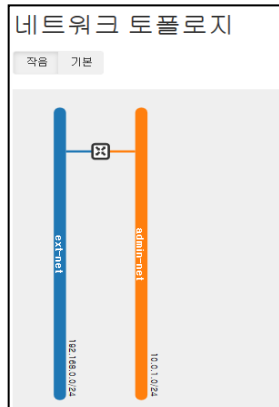
admin-net: 10.0.1.0/24 (admin-subnet) 설명:
라우터에 지정된 서브넷을 연결할 수 있습니다.

IP 주소 (옵션) 설명:
생성된 인터페이스의 기본 IP 주소는 선택된 서브넷의 게이트웨이입니다. 인터페이스의 다른 IP 주소를 지정할 수 있습니다. 위의 목록 속하는 지정된 IP 주소를 서브넷에서 선택해야 합니다.

라우터 이름 *

라우터 ID *

취소 인터페이스 추가



게이트웨이 설정

외부 네트워크 *

ext-net 설명:
라우터에 지정된 외부 네트워크를 연결할 수 있습니다. 외부 네트워크는 라우터의 기본 경로와 외부 연결을 위한 게이트웨이 역할을 합니다.

라우터 이름 *

라우터 ID *

취소 게이트웨이 설정

neutron router-gateway-set admin-router ext-net

Set gateway for router admin-router

Network만들기

Vm생성

neutron router-interface-add admin-router admin-subnet

시큐리티 그룹 룰 관리:default

시큐리티 그룹 룰

Direction	Ether 타입	IP 프로토콜	포트 범위	원격	작업
내보냄	IPv4	이디온	-	0.0.0.0 (CIDR)	공식
내보냄	IPv6	이디온	-	:::0 (CIDR)	공식
들어옴	IPv6	이디온	-	default	공식
들어옴	IPv4	이디온	-	default	공식
들어옴	IPv4	ICMP	-	0.0.0.0 (CIDR)	공식
들어옴	IPv4	TCP	22 (SSH)	0.0.0.0 (CIDR)	공식
들어옴	IPv4	TCP	80 (HTTP)	0.0.0.0 (CIDR)	공식

시큐리티 그룹 키 패어

키 패어 이름	Fingerprint	작업
keypair	42:3b:01:76:a4:4c:5c:d4:70:65:86:2e:89:2d:a7:6f	키 패어 삭제
test-key	0a:ca:b2:d3:f9:92:42:71:9e:4a:b8:8f:c5:49:60:61	키 패어 삭제

시큐리티 그룹 유동 IP

IP 주소	고정 IP 주소 범위	유동 IP Pool	작업
192.168.0.202	test_instance2 10.0.1.4	ext-net	연결 해제

인스턴스

인스턴스

인스턴스 이름 필터

인스턴스 이름	이미지 이름	IP 주소	크기	키 패어	상태	가용성 존	작업	전원 상태	생성된 이후 시간	작업
test_instance2	cirros-0.3.3-x86_64	10.0.1.4 192.168.0.202	m1.tiny	keypair	작업	nova	None	실행	4 minutes	스냅샷 생성

cirros 등록 정보

분류(C):

- 연결
 - 사용자 인증
 - 프로토콜
 - 로그인 스크립트
 - SSH
 - 보안
 - 터널링
 - SFTP
 - TELNET
 - RLOGIN
 - SERIAL
 - 터미널
 - 키보드
 - VT 모드
 - 고급
 - 모양
 - 여백
 - 고급
 - 추적
 - 로그
 - ZMODEM

연결 > 사용자 인증

사용자 인증 방법과 관련 내용을 지정하십시오. 이 페이지는 로그인 과정의 편의를 위해서 제공됩니다. 보안이 중요한 호스트에 접속하는 경우에는 이 페이지에 사용자 인증과 관련된 아무 정보도 입력하지 않는 편이 좋습니다.

인증 방법(M): Public Key

사용자 이름(U): cirros

비밀 번호(P):

사용자 키(K): keypair

암호(E):

참고: Public Key 및 Keyboard Interactive 사용자 인증은 SSH/SFTP 프로토콜에서만 사용 가능합니다.

확인 취소

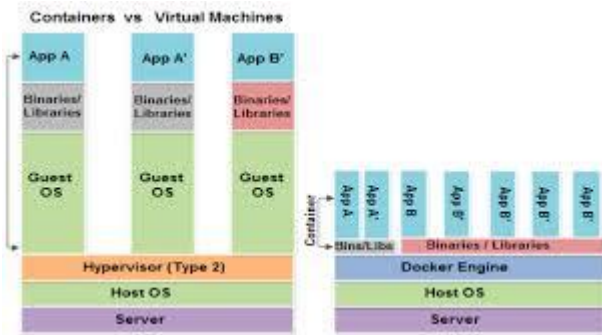
Contents

1. Openstack 인프라 구축 (4 node 구성) [30분]
2. Openstack 위에 VM 생성 [20분]
3. **docker** 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. Docker로 WEB서비스 구축 [15분]
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. Docker로 jenkins 구현 [30분]

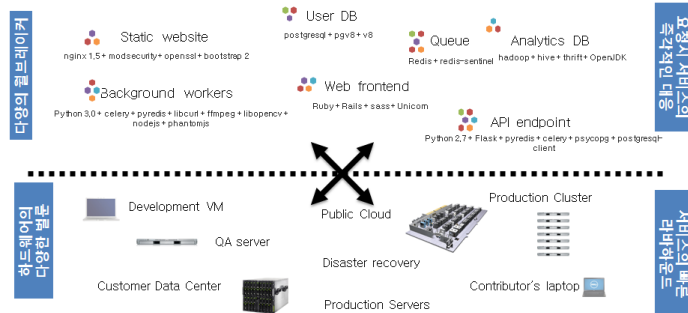
What to do during 30 min.

Docker / Container / Openstack / CI

- Docker는 개발자와 시스템 관리자가 어플리케이션을 개발하고, 배포하고, 운영하는 플랫폼
- Docker는 이미지 기반(애플리케이션과 그 환경까지를 모두 포함) / 그 이미지를 기반으로 컨테이너 형태로 구현
- 30분 동안 우리가 할 것 (시스템 인스톨 / 애플리케이션 구축 / 구성 / 도커 이미지 관리 / 업로드)



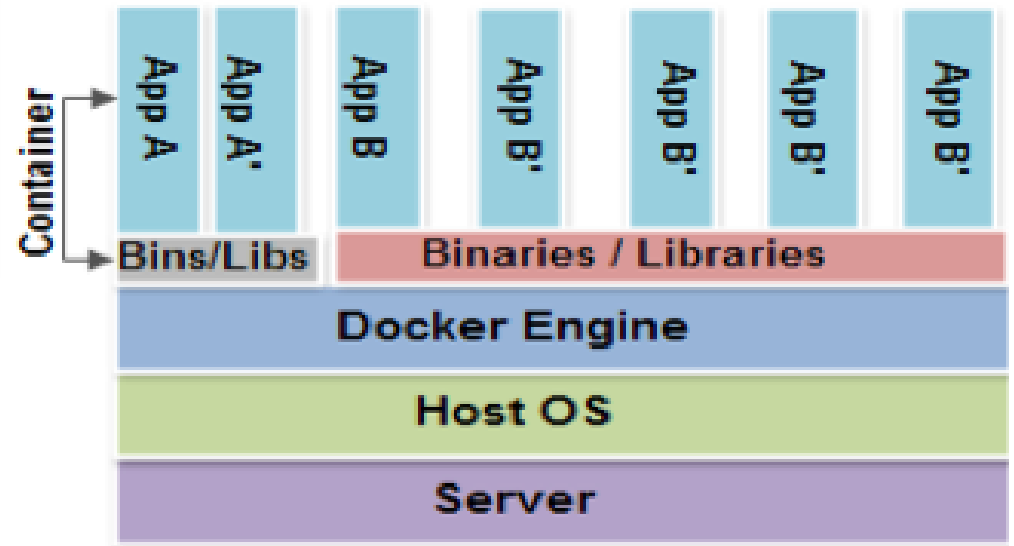
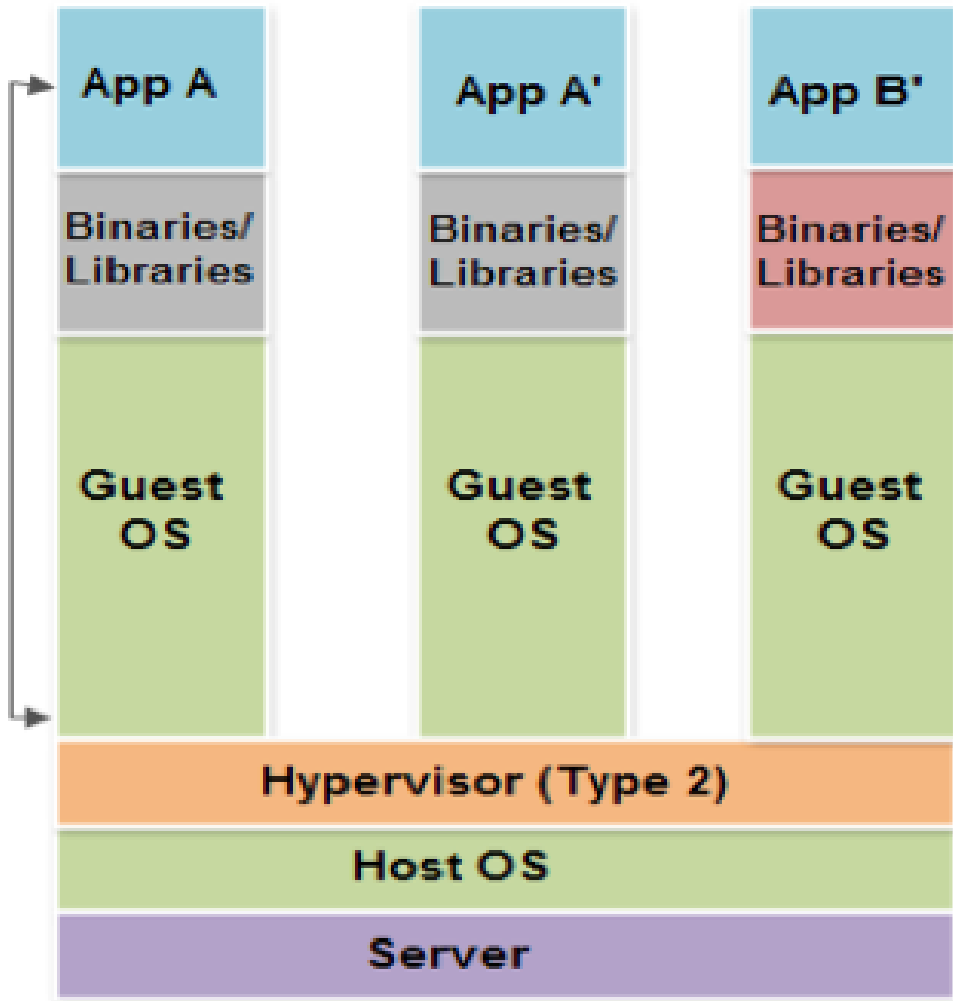
The Challenge



Slide Reference: DockerCon




Containers vs Virtual Machines



The Challenge

다양한 웹프레임워크


요청시 서비스의
응답 시간 최소화

 Static website
nginx 1.5 + modsecurity + openssl + bootstrap 2

 User DB
postgresql + pgv8 + v8

 Queue
Redis + redis-sentinel

 Analytics DB
hadoop + hive + thrift + OpenJDK

 Background workers
Python 3.0 + celery + pyredis + libcurl + ffmpeg + libopencv +
nodejs + phantomjs

 Web frontend
Ruby + Rails + sass + Unicorn

 API endpoint
Python 2.7 + Flask + pyredis + celery + pycopg + postgresql-
client



하드웨어의
다양한 플랫폼

서비스의 빠른
리버추얼

 Development VM

 QA server

Customer Data Center 

Public Cloud


Disaster recovery

Production Servers



Production Cluster



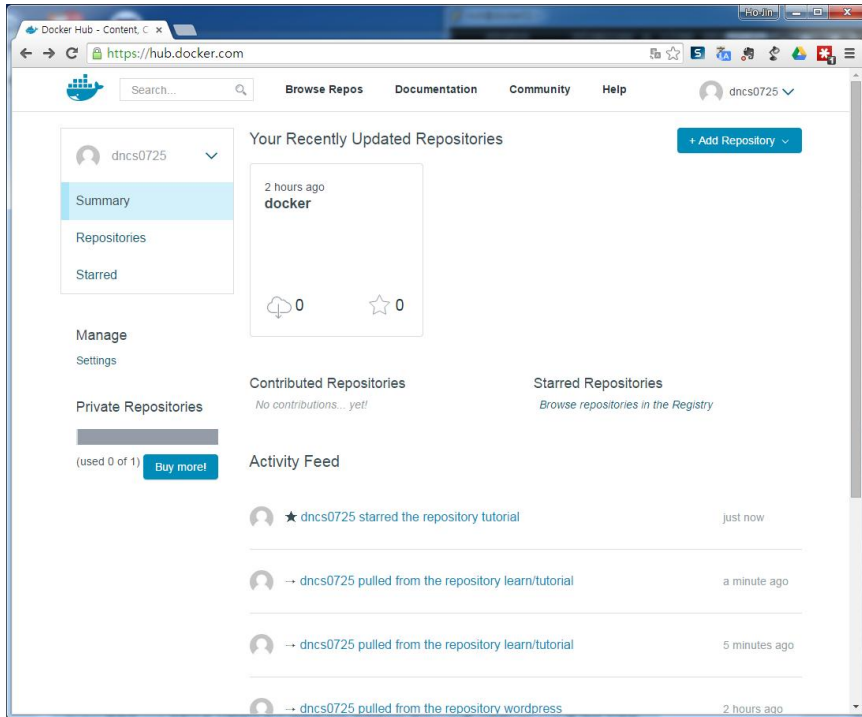
Contributor's laptop 

Slide Reference: DockerCon

Getting Started with Docker Hub.

Docker Hub

- 먼저 docker site에 가입 (history 관리)
- Hub.docker.com



```
[root@Tiger ~]# docker login
Username: oscinfra
Password:
Email: khoj@osci.kr
Login Succeeded
```

Installing Docker on CentOS 7

- Hostname setting
- Docker install / Docker start 등록 / 최신버전 update

```
#env setting
export LANG=en_US.UTF-8;

#hostname setting
yes | cp -f /dev/null /etc/hostname;echo "docker" > /etc/hostname ; hostname;
systemctl disable firewalld; systemctl stop firewalld

#repo copy
scp 192.168.0.220:/etc/yum.repos.d/osc.repo /etc/yum.repos.d/osc.repo

#docker installation/setting
yum -y install update > /dev/null ; yum -y install docker; systemctl enable docker;
systemctl start docker; systemctl status docker | grep Active

#update docker to latest version
#wget https://get.docker.com/builds/Linux/x86_64/docker-latest -O $(type -P docker)

# check the version
docker version
```

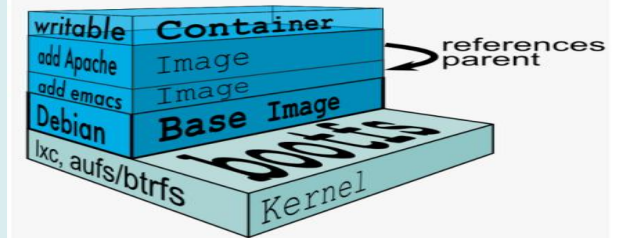
Downloading container images & Running Docker Containers

Docker 로 할 수 있는 일

- Image 가지고 오기
- 새로운 container 만들기
- 파일시스템 할당하여, read-wirte layer 마운트 하기
- 네트워크 브릿지 인터페이스 할당하기
- IP 주소 할당하기
- 프로세스 실행하기
- 프로세스/어플리케이션 output 보여주기

Docker command

- `docker ps -`
- `docker logs -`
- `docker stop -`
- `docker build`
- `docker commit`
- `docker cp -`
- `docker diff -`



```
[root@juno-compute yum.repos.d]# docker search centos
```

NAME	DESCRIPTION	STARS	
OFFICIAL	AUTOMATED		
centos	The official build of CentOS.	864	[OK]

```
[root@juno-compute ~]# docker pull centos
Pulling repository centos
88f9454e60dd: Download complete
Status: Image is up to date for centos:latest
```

```
[root@juno-compute ~]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
centos	7	88f9454e60dd	4 days ago	223.9 MB
centos	centos7	88f9454e60dd	4 days ago	223.9 MB
centos	latest	88f9454e60dd	4 days ago	223.9 MB

```
[root@juno-compute ~]# docker run centos /bin/echo "Hello World"
Hello World
```

Downloading container images & Running Docker Containers

Docker info

```
[root@docker /]# docker info
Containers: 6
Images: 18
Storage Driver: devicemapper
  Pool Name: docker-253:1-67220484-pool
  Pool Blocksize: 65.54 kB
  Data file:
  /var/lib/docker/devicemapper/devicemapper/data
  Metadata file:
  /var/lib/docker/devicemapper/devicemapper/metadata
  Data Space Used: 831.9 MB
  Data Space Total: 107.4 GB
  Metadata Space Used: 1.696 MB
  Metadata Space Total: 2.147 GB
  Library Version: 1.02.84-RHEL7 (2014-03-26)
Execution Driver: native-0.2
Kernel Version: 3.10.0-123.20.1.el7.x86_64
Operating System: CentOS Linux 7 (Core)
```

```
[root@docker ~]# docker info
Containers: 15
Images: 43
Storage Driver: devicemapper
  Pool Name: docker-253:1-67220484-pool
  Pool Blocksize: 65.54 kB
  Data file:
  /var/lib/docker/devicemapper/devicemapper/data
  Metadata file:
  /var/lib/docker/devicemapper/devicemapper/metadata
  Data Space Used: 2.682 GB
  Data Space Total: 107.4 GB
  Metadata Space Used: 4.174 MB
  Metadata Space Total: 2.147 GB
  Library Version: 1.02.84-RHEL7 (2014-03-26)
Execution Driver: native-0.2
Kernel Version: 3.10.0-123.20.1.el7.x86_64
Operating System: CentOS Linux 7 (Core)
Username: oscinfra
Registry: [https://index.docker.io/v1/]
```


Run your new image [Dockerizing Applications]

: A "Hello world" / An Interactive Container

- 컨테이너를 상자 안의 프로세스라고 비유할 수 있습니다.
- 상자는 프로세스가 필요로 하는 모든 것, 파일 시스템, 시스템 라이브러리, 셸 등을 가지고 있지만, 기본적으로 아무것도 실행하고 있지 않습니다.
- 그 프로세스들을 실행함으로써, 컨테이너를 시작합니다.

```
[root@docker ~]# docker run centos echo "Hellow world"
Hellow world
```

```
[root@docker01 ~]# docker run -i -t --name osc centos /bin/bash # i,--
interacitve . -t,--tty
```

```
[root@6cfe2306796b /]# ip a | grep eth0
11: eth0: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    inet 172.17.0.4/16 scope global eth0
[root@6cfe2306796b /]# yum install -y wget > /dev/null ; exit
[root@docker ~]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
6cfe2306796b	centos:7	"/bin/bash"	5 minutes ago	Exited (0) 46 seconds ago		osc

```
[root@docker ~]# docker start osc ;docker ps ; docker attach osc
osc
[root@6cfe2306796b /]# rpm -qa wget
wget-1.14-10.el7_0.1.x86_64
[root@6cfe2306796b /]# ctrl+p , ctrl+q # shell 종료없이 호스트로 복귀
```

Run your new image [Dockerizing Applications]

A Daemonized Hello world

- 컨테이너를 데몬처럼 돌리기

```
[root@docker ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
centos               7                  88f9454e60dd       2 days ago        223.9 MB
centos               centos7            88f9454e60dd       2 days ago        223.9 MB
centos               latest             88f9454e60dd       2 days ago        223.9 MB

[root@docker ~]# docker run --name osc -d centos /bin/bash \
-c "while true;do echo Hello world;sleep 1;done"
61fe6bea107205c3ba9bfe998e506297797f0491d6bbe32664f5db261641c5ee

[root@docker01 ~]# ps -ef | grep -v grep | grep true
root      5338   3689   0 16:18 ?        00:00:00 /bin/bash -c while true;do echo Hello world;sleep 1;done
root      5357   3525   0 16:18 pts/0    00:00:00 grep --color=auto true

[root@docker ~]# docker logs --tail=10 -ft osc
2015-03-07T11:07:18.655548350Z Hello world

[root@docker ~]# docker top osc
UID          PID          PPID         C           STIME      TTY      TIME          CMD
root         3408         983          0           06:02      ?        00:00:00    /bin/bash -c
while true;do echo Hello world;sleep 1;done
root         3827         3408         0           06:08      ?        00:00:00    sleep 1

[root@docker ~]# docker inspect osc / docker stop osc / docker kill osc
```

Think/Think/Think

We met some errors as below / where is the docker definition file?

- [root@docker01 ~]# docker run -i -t --name test learn/tutorial /bin/bash
FATA[0000] Error response from daemon: Conflict. The name "test" is already in use by container c4803f88e5c4. You have to delete (or rename) that container to be able to reuse that name.

```
[root@docker01 ~]# docker rm c4803f88e5c4
c4803f88e5c4
[root@docker01 ~]# docker run -i -t --name test learn/tutorial /bin/bash
root@1dbb3af8ec20:/#
```

- Where is the docker definition file?

```
[root@docker ~]# docker inspect osc # docker inspect -f '{{.Name}}'
[{"
  "AppArmorProfile": "",
  "Args": [
    "-c",
    "while true;do echo Hello world;sleep 1;done" ]

#images /var/lib/docker/conatiner
[root@docker docker]# cd conatainers/61fe6bea107205c3ba9bfe9*
61fe6bea107205c3ba9bfe998e506297797f0491d6bbe32664f5db261641c5ee]# ls
61feXX-json.log config.json hostconfig.json hostname hosts resolv.conf
secrets
```

Running a Web Application in Docker

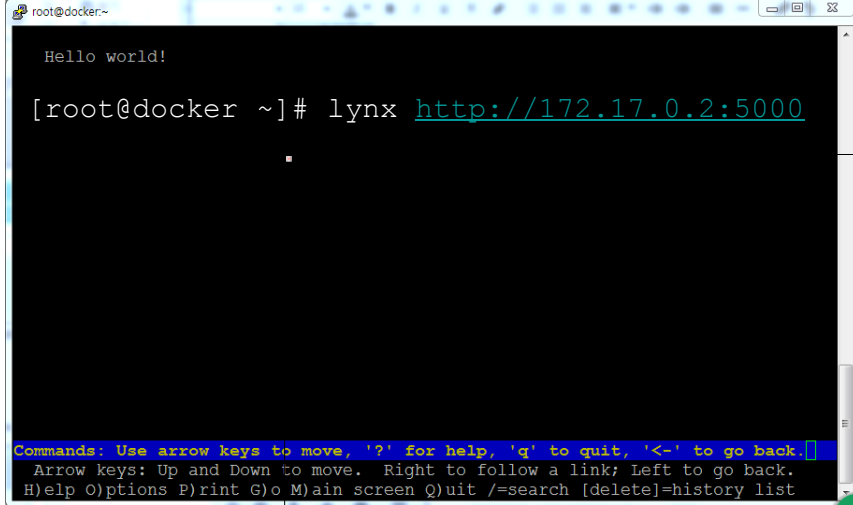
training/webapp

```
[root@docker01 ~]# docker run -d -P training/webapp python app.py # -d, --
detach=false /
[root@docker01 ~]# docker ps -l
CONTAINER ID          IMAGE                COMMAND              CREATED
STATUS               PORTS              NAMES
489160dbba6b        training/webapp:latest  "python app.py"     5 minutes
ago                 Up 5 minutes      0.0.0.0:49154->5000/tcp  serene_heisenberg
[root@docker01 ~]# docker logs -f cocky_hopper
* Running on http://0.0.0.0:5000/
192.168.0.4 - - [28/Feb/2015 22:14:54] "GET / HTTP/1.1" 200 -
192.168.0.4 - - [28/Feb/2015 22:14:55] "GET /favicon.ico HTTP/1.1" 404 -
```

```
# ip addr show docker0:
inet 172.17.42.1/16 scope docker0
```

```
[root@docker ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source
destination
```

```
ACCEPT     tcp  --  anywhere
172.17.0.2      tcp
dpt:complex-main
```



```
root@docker:~
Hello world!
[root@docker ~]# lynx http://172.17.0.2:5000
.
Commands: Use arrow keys to move, '?' for help, 'q' to quit, '<-' to go back.
Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

Running a Web Application in Docker

Inspect training/webapp

```
[root@docker01 ~]# docker top cocky_hopper
UID                PID                PPID              C
STIME             TTY                TIME             CMD
root              7544              3689             0
17:12             ?                  00:00:00         python app.py
[root@docker01 ~]# docker inspect cocky_hopper
"Cmd": [
    "python",
    "app.py"
  "ExposedPorts": {
    "5000/tcp": {}
  "Hostname": "999cdee2c894",
  "Image": "training/webapp",
"Name": "/cocky_hopper",
  "NetworkSettings": {
    "Gateway": "172.17.42.1",
    "IPAddress": "172.17.0.19",
    "IPPrefixLen": 16,
    "Ports": {
      "5000/tcp": [
        "HostIp": "0.0.0.0",
        "HostPort": "49154"
```

Managing Data in Containers

Mount a Host Directory as a Data Volume

```
[root@docker01 ~]# docker run -d -P --name web -v /webapp training/webapp
python app.py
# -v, --volume=[] Bind mount a volume (e.g., from the host: -v
/host:/container, from Docker: -v /container)
191388a413d843a9e6ae020b9bf051698b8755e7081e2d9eeab77a2dbb72bdd1
=====

[root@docker ~]# docker run -d -P --name web -v /src/webapp:/opt/webapp
training/webapp python app.py

[root@docker ~]# cd /src;ls
webapp
```

Mount a Host File as a Data Volume

```
[root@docker]# docker run --rm -it -v ~/.bash_history:/.bash_history centos
/bin/bash

[root@5bf8bf23f10b /]# ls -al | more
ls: cannot access .bash_history: Permission denied
-??????????? ? ? ? ? ? .bash_history
```

Making images (2 methods)

From container

```
[root@docker01 ~]# docker run -t -i training/sinatra /bin/bash
Status: Downloaded newer image for training/sinatra:latest
root@62f680cfd5a4:/# gem install json ;exit
Fetching: json-1.8.2.gem (100%)
exit
[root@docker01 ~]# docker ps -l
CONTAINER
ID                IMAGE                COMMAND                CREATED                STATUS
62f680cfd5a4     training/sinatra:latest  "/bin/bash"          9 minutes ago        Exited (0) About a minute ago
angry_yonath

[root@docker01 ~]# docker commit -m "Added json gem" -a "Kate
Smith" 62f680cfd5a4 ouruser/sinatra:v2
52fc4cf3a3dc049ecd43f0626b53c4480305f8463461bd519c338f99a4c2743b

[root@docker01 ~]# docker images
REPOSITORY        TAG                IMAGE ID                CREATED                VIRTUAL SIZE
ouruser/sinatra   v2                52fc4cf3a3dc          About a minute ago    451.9 MB
[root@docker01 ~]# docker run -t -i ouruser/sinatra:v2 /bin/bash
root@215d1f67558b:/#
```

Making images (2 methods)

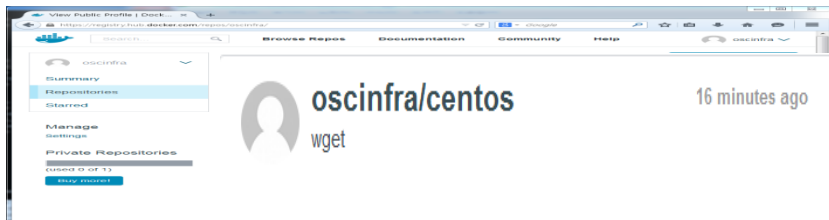
From Dockerfile

```
[root@docker wget]# cat Dockerfile
# for the technet seminar by hojin kim
FROM oscinfra/centos:tool
MAINTAINER hojin kim "khoj@osci.kr"
RUN yum install -y wget
RUN mkdir /root/wget
EXPOSE 22 # default port
```

```
[root@docker]# docker images
```

REPOSITORY	TAG	VIRTUAL SIZE	IMAGE ID
oscinfra/centos	intall_wgeet		
c945ac8f8743	19 seconds ago	379.9 MB	
oscinfra/centos	tool		
1f06057f9152	24 minutes ago	366.5 MB	
oscinfra/centos	latest		
403871a8320a	26 minutes ago	366.5 MB	

```
[root@docker ~]# docker push
oscinfra/centos:tool
The push refers to a repository
[oscinfra/centos] (len: 1)
Sending image list
Pushing repository oscinfra/centos (1 tags)
```



```
[root@docker wget]# docker build -
t="oscinfra/centos:intall_wget" .
Sending build context to Docker daemon 2.56 kB
Sending build context to Docker daemon
```

```
Step 0 : FROM oscinfra/centos:latest
---> 403871a8320a
Step 1 : MAINTAINER hojin kim "khoj@osci.kr"
---> Running in 4c4bc393c67e
---> 8cc5127c853a
Removing intermediate container 4c4bc393c67e
Step 2 : RUN yum install -y wget
---> Running in 2ca7b10b283a
Loaded plugins: fastestmirror

Installed:
  wget.x86_64 0:1.14-10.el7_0.1
```

```
Complete!
---> 3bbded5a9761
Removing intermediate container 2ca7b10b283a
Step 3 : RUN mkdir /root/wget
---> Running in 2de6060b4562
---> 6ba1987b89a7
Removing intermediate container 2de6060b4562
Step 4 : EXPOSE 22
---> Running in 59d051bb382d
---> c945ac8f8743
Removing intermediate container 59d051bb382d
Successfully built c945ac8f874
```


Docker Compose

Docker compose 구성 예 (뒤에 자세히 나옴)

February 26, 2015

ANNOUNCING DOCKER COMPOSE

Today we're excited to announce that Docker Compose is available for download. Docker Compose is an orchestration tool that makes spinning up multi-container applications effortless. [Head to the install docs to download it.](#)

With Compose, you define your application's components – their containers, their configuration, links, volumes, and so on – in a single file, then you can spin everything up with a single command that does everything that needs to be done to get your application running.

If you've used [Fig](#) before, this will sound familiar – in fact, Compose is based directly on the Fig codebase and is backwards-compatible with Fig applications. Fig's been hugely successful as a tool for development environments, with almost 5,000 stars on GitHub, 80,000 downloads, and users including Yelp, Spotify, Mozilla, Facebook and the UK Government. Fig will continue to receive critical maintenance updates, but is now deprecated in favour of Compose – check the [release notes](#) for how to upgrade (spoiler: it's very simple).

```
you@tutorial:~$ cat docker-compose.yml
```

```
web:
  build: .
  links:
    - db
  ports:
    - "8000:8000"
db:
  image: postgres
```

```
you@tutorial:~$ cat Dockerfile
```

```
FROM python:2.7
WORKDIR /code
ADD requirements.txt /code/
RUN pip install -r requirements.txt
ADD . /code
CMD python app.py
```

```
[root@docker01 wordpress]# docker-compose up
[root@docker01 ~]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS	NAMES	
7ddf0f0dec20	wordpress_web:latest	"php -S 0.0.0.0:8000	
4 minutes ago	Up 4 minutes	0.0.0.0:8000->8000/tcp	
wordpress_web_1			
83ddc6ea784c	orchardup/mysql:latest	"/usr/local/bin/run"	
4 minutes ago	Up 4 minutes	3306/tcp	
wordpress_db_1			

Contents

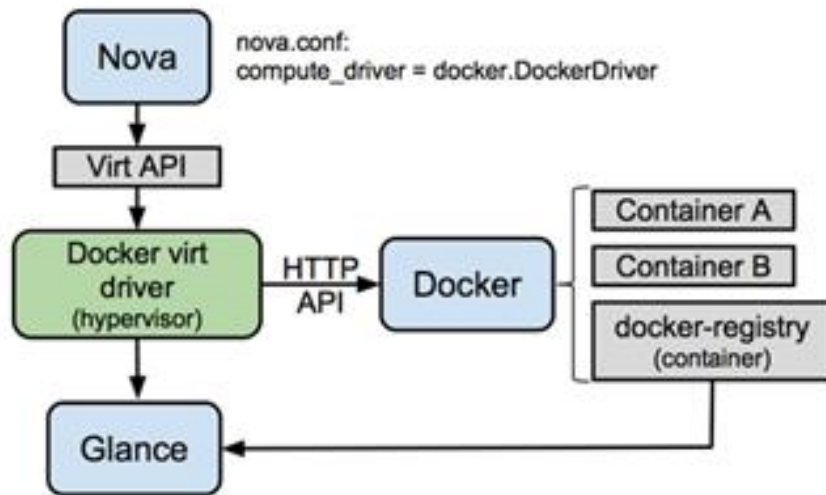
1. Openstack 인프라 구축 (4 node 구성) [30분]
2. Openstack 위에 VM 생성 [20분]
3. docker 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. Docker로 WEB서비스 구축 [15분]
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. Docker로 jenkins 구현 [30분]

What to do during 20 min.

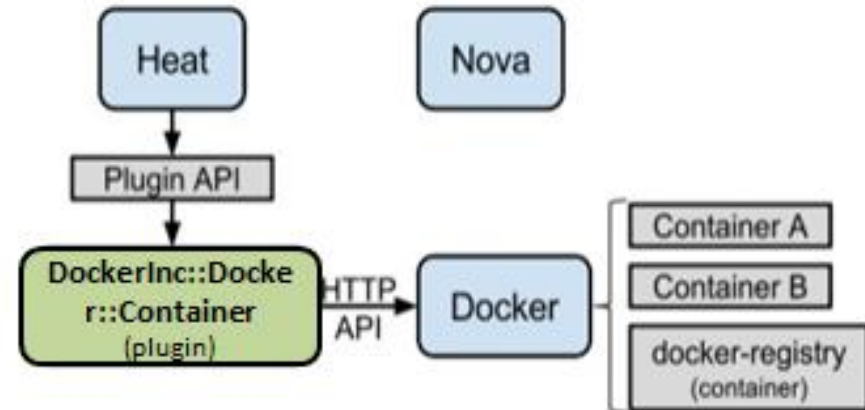
Docker + openstack

- Docker와 openstack의 연동 필요성
- Docker와 openstack을 연동하기

nova-docker virt driver



docker heat plugin



Connect to Openstack

Docker prerequisite

- Install python-pip/git/gcc/wget/lynx
- Install Oslo.log

```
yum install -y python-pip git gcc wget
yum install -y docker
usermod -G docker nova
service openstack-nova-compute restart
pip install pbr
wget https://pypi.python.org/packages/source/o/oslo.log/oslo.log-0.4.0.tar.gz#md5=e02b6feebe849c8bae50b5c329f7a9e0
tar -xvf oslo.log-0.4.0.tar.gz
cd ./oslo.log-0.4.0
python setup.py install ; pip install pbr
```

Nova-docker

- Install nova-docker
- git checkout stable/juno

```
git clone https://github.com/stackforge/nova-docker.git
cd nova-docker/

git checkout stable/juno
python setup.py install
```

Change openstack setting

Setting env

- `chmod 666 /var/run/docker.sock`
- `mkdir /etc/nova/rootwrap.d`

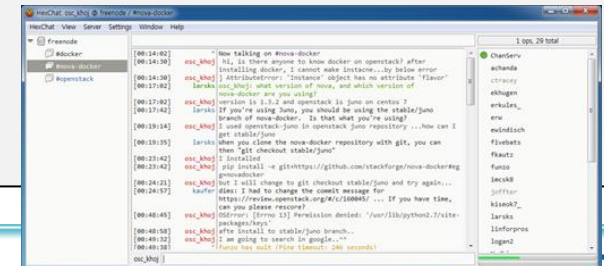
```
cat /etc/nova/rootwrap.d/docker.filters
# nova-rootwrap command filters for setting up network in the docker driver
# This file should be owned by (and only-writeable by) the root user
[Filters]
# nova/virt/docker/driver.py: 'ln', '-sf', '/var/run/netns/.*'
ln: CommandFilter, /bin/ln, root
```

```
service docker start
chmod 660 /var/run/docker.sock
mkdir /etc/nova/rootwrap.d
```

```
cat /etc/nova/nova.conf # compute dirver 바꾸기
compute_driver = novadocker.virt.docker.DockerDriver
```

```
cat /etc/glance/glance-api.conf # container 형식으로 지원변경
container_formats=ami,ari,aki,bare,ovf,ova,docker
```

```
service openstack-glance-api restart
service openstack-nova-compute restart
```



Make glance image

Setting env

- check docker
- Check the openstack

```
$ docker pull busybox
$ docker save busybox | glance image-create --is-public=True --container-
format=docker --disk-format=raw --name busybox
nova keypair-add mykey > mykey.pem
nova boot --flavor m1.small --image cirros --key-name mykey test1
nova list
ssh -i ../devstack/mykey.pem cirros@<IP ADDRESS>
```

```
docker pull busybox:latest
cd source keystone_admin
docker save busybox | glance image-create --is-public=True --container-
format=docker --disk-format=raw --name busybox
glance image-list
nova boot --image busybox --flavor m1.tiny --nic net-id a937454d-a905-
43d2-818d-8fc5a920d8f2 busybox
docker ps -a
docker attach <CONTAINER ID from command above>
```

Check the status

Setting env

- Docker 상태를 먼저 살펴본다.

```
[$[root@juno-compute nova-docker]# docker run -i -t fedora /bin/bash
Pulling repository fedora
834629358fe2: Download complete
Status: Downloaded newer image for fedora:latest
bash-4.3#
```

- 간단한 이미지를 만들어본다.

```
[root@juno-compute nova-docker]# docker pull larsks/thttpd
Pulling repository larsks/thttpd
a32a10d723ab: Download complete
Status: Downloaded newer image for larsks/thttpd:latest
[root@juno-compute ~]# docker save larsks/thttpd | glance image-create -
-name larsks/thttpd --container-format docker --disk-format raw --is-
public true
```

Property	Value
checksum	cc8b32dcc9d12fbfa1d59ce655457d31
name	larsks/thttpd

Check the status

Gui 화면에서 만들어 보기

```
[root@juno-compute ~]# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED
148ef2905e65	larsks/thttpd:latest	"/thttpd -D -l /dev/	About a minute ago
Up About a minute		nova-f1aeb1e3-d395-4138-a92e-73c77e854709	
b122f9046020	larsks/thttpd:latest	"/thttpd -D -l /dev/	2 minutes ago
Up 2 minutes		nova-ac8d4a33-776b-4a13-be49-6b8bcfa87ec6	
e8dc72cd6a65	larsks/thttpd:latest	"/thttpd -D -l /dev/	9 minutes ago
Up 9 minutes		nova-d16b6bfe-4daa-48e5-a790-a9be088412ac	

Instances

Instances

Instance Name Filter Filter

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
<input type="checkbox"/>	test03	larsks/thttpd	10.0.1.7 192.168.0.204	m1.tiny	keypair	Active	nova	None	Running	0 minutes	Create Snapshot <input type="button" value="v"/>
<input type="checkbox"/>	test02	larsks/thttpd	10.0.1.6 192.168.0.202	m1.tiny	keypair	Active	nova	None	Running	1 minute	Create Snapshot <input type="button" value="v"/>
<input type="checkbox"/>	test01	larsks/thttpd	10.0.1.5 192.168.0.203	m1.tiny	test-key	Active	nova	None	Running	9 minutes	Create Snapshot <input type="button" value="v"/>

Displaying 3 items

사각형 캡처(R)

Contents

1. Openstack 인프라 구축 (4 node 구성) [30분]
2. Openstack 위에 VM 생성 [20분]
3. docker 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. **Docker로 WEB서비스 구축 [15분]**
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. Docker로 jenkins 구현 [30분]

What to do during 30 min.

Install MySQL and WordPress Using Docker

- Mysql과 wordpress 로 부터 latest docker image 가지고 오기
- 간단한 setting으로 올리기
- 파일 수정후 image생성

The image displays three overlapping screenshots of the WordPress installation process:

- Left Screenshot:** Shows the language selection screen. The URL is `192.168.0.181/wp-admin/install.php`. A dropdown menu is open, showing various languages, with "English (United States)" selected. A "Continue" button is visible at the bottom.
- Middle Screenshot:** Shows the "Welcome" and "Information needed" section. The URL is `192.168.0.181/wp-admin/install.php?step=1`. The site title is "docker", the username is "hojin kim", and the email is "khoj@osci.kr". The password is marked as "Very weak". A "Privacy" checkbox is checked, and an "Install WordPress" button is at the bottom.
- Right Screenshot:** Shows the "Log In" screen. The URL is `192.168.0.181/wp-login.php`. The username field contains "khoj@osci.kr". There is a "Log In" button and a "Remember Me" checkbox.

What to do during 30 min.

Install MySQL and WordPress Using Docker

- Mysql / wordpress image 가지고 오기

```
[root@docker01 wordpress]# docker pull mysql:latest
Pulling repository mysql
0beee7f478c8: Download complete

Status: Downloaded newer image for mysql:latest
[root@docker01 wordpress]# docker images
REPOSITORY          TAG          IMAGE ID          CREATED          VIRTUAL SIZE
mysql               latest      0beee7f478c8     3 weeks ago     283.1 MB
[root@docker01 wordpress]# docker run --name test-mysql -e MYSQL_ROOT_PASSWORD='test'
-d mysql
8f0c7c57b35a4177e8722d4868ff891a2821776dd3f075e02a126475f94f41db
[root@docker01 wordpress]#
[root@docker01 wordpress]# docker ps -l
CONTAINER ID        IMAGE          COMMAND          CREATED
STATUS            PORTS         NAMES
8f0c7c57b35a      mysql:5       "/entrypoint.sh mysq   28 seconds ago   Up
27 seconds       3306/tcp     test-mysql
```

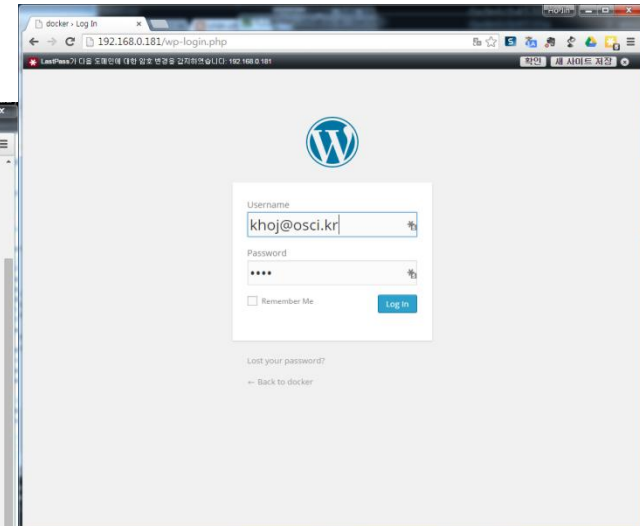
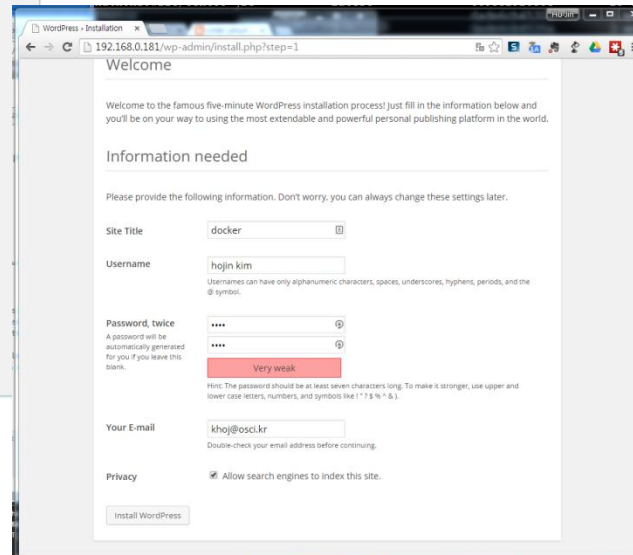
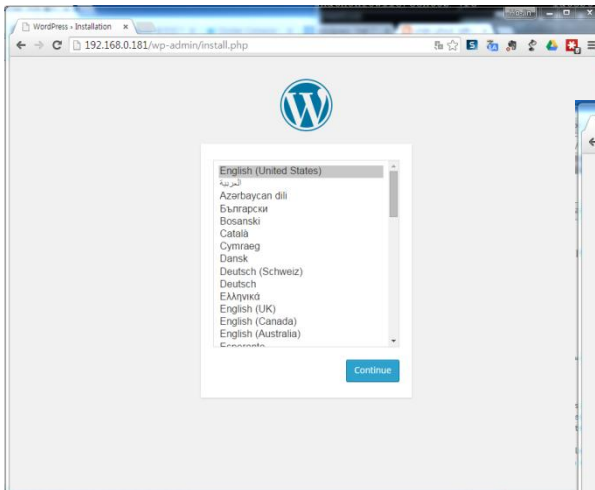
What to do during 30 min.

Install MySQL and WordPress Using Docker

Wordpress 실행

```
[root@docker01 wordpress]# docker pull wordpress:latest
Pulling repository wordpress
048e0caa51ac: Pulling dependent layers
511136ea3c5a: Download complete
Status: Downloaded newer image for wordpress:latest
```

```
[root@docker01 wordpress]# docker run --name test-wordpress --link test-mysql:mysql -p 80:80 -d
wordpress
```



What to do during 30 min.

Install MySQL and WordPress Using Docker

- 수정 후 이미지 commit하기

```
[root@docker01 wordpress]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
981485e03a06        wordpress:4        "/entrypoint.sh apac 25 minutes ago    Up
p 25 minutes        0.0.0.0:80->80/tcp  test-wordpress
8f0c7c57b35a        mysql:5           "/entrypoint.sh mysq 32 minutes ago    Up
p 32 minutes        3306/tcp          test-mysql
[root@docker01 wordpress]# docker commit 9814 osc_wordpress
-bash: docker: command not found
[root@docker01 wordpress]# docker commit 9814 osc_wordpress
10a6ed19d49729deeacf5c9bb6c91a6f0ef17c08d04664c6294e64b49b2c09d6
[root@docker01 wordpress]# docker commit 8f0c osc_mysql
C6b9d3708309f98304c547ab2d39924dc39741b7b62d9252ed062c252041554a

//docker stop docker stop `docker ps -qa`

[root@docker01 ~]# docker run --name osc_mysql -e MYSQL_ROOT_PASSWORD='test' -d
osc_mysql
fb2e1ee783747dbf9c77f35648752a29735a982bf3bdb07cfdb7631b05a8bd28

docker run --name osc_wordpress2 --link test-mysql:mysql -p 80:80 -d osc_wordpress
```

Getting started with Compose and Wordpress

Install MySQL and WordPress Using Docker

Docker-compose 구성

```
[root@docker01 dbdata]# curl -L
https://github.com/docker/compose/releases/download/1.1.0/docker-compose-`uname -s`-
`uname -m` > /usr/local/bin/docker-compose
  % Total      % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100   407      0   407    0     0    376      0  --:--:--  0:00:01 --:--:--   376
100 5134k   100 5134k    0     0 1135k      0  0:00:04  0:00:04 --:--:-- 1760k
[root@docker01 dbdata]# chmod +x /usr/local/bin/docker-compose
[root@docker01 dbdata]# docker-compose --version
docker-compose 1.1.0
```

```
[root@docker01 ~]# curl https://wordpress.org/latest.tar.gz | tar -xvzf -
  % Total      % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
  0 6041k    0 3726    0     0 4094      0  0:25:11  --:--:--  0:25:11 4094wordpress/
wordpress/wp-settings.php
```

Getting started with Compose and Wordpress

Install MySQL and WordPress Using Docker

- Dockerfile 구성과 compose 파일 구성

```
[root@docker01 wordpress]# cat Dockerfile
FROM orchardup/php5
ADD . /code
[root@docker01 wordpress]# cat docker-compose.yml
web:
  build: .
  command: php -S 0.0.0.0:8000 -t /code
  ports:
    - "8000:8000"
  links:
    - db
  volumes:
    - ./code
db:
  image: orchardup/mysql
  environment:
    MYSQL_DATABASE: wordpress
```


Getting started with Compose and Wordpress

Install MySQL and WordPress Using Docker

간단 program

```
[root@docker01 wordpress]# cat wp-config.php
<?php
define('DB_NAME', 'wordpress');
define('DB_USER', 'root');
define('DB_PASSWORD', '');
define('DB_HOST', "db:3306");
define('DB_CHARSET', 'utf8');
define('DB_COLLATE', '');

define('AUTH_KEY',          'test');
define('SECURE_AUTH_KEY',  'test');
define('LOGGED_IN_KEY',    'test');
define('NONCE_KEY',        'test');
define('AUTH_SALT',         'test');
define('SECURE_AUTH_SALT', 'test');
define('LOGGED_IN_SALT',   'test');
define('NONCE_SALT',        'test');

$table_prefix = 'wp_';
define('WPLANG', '');
define('WP_DEBUG', false);

if ( !defined('ABSPATH') )
    define('ABSPATH', dirname(__FILE__) . '/');

require_once(ABSPATH . 'wp-settings.php');
```

```
[root@docker01 wordpress]# cat router.php
<?php
$root = $_SERVER['DOCUMENT_ROOT'];
chdir($root);
$path =
    '/' . ltrim(parse_url($_SERVER['REQUEST_URI'])['path'], '/');
set_include_path(get_include_path() . ':' . __DIR__);
if (file_exists($root.$path))
{
    if (is_dir($root.$path) &&
        substr($path, strlen($path) - 1, 1) !== '/')
        $path = rtrim($path, '/') . '/index.php';
    if (strpos($path, '.php') === false) return
        false;
    else {
        chdir(dirname($root.$path));
        require_once $root.$path;
    }
} else include_once 'index.php';
```

Getting started with Compose and Wordpress

Install MySQL and WordPress Using Docker

실행

```
[root@docker01 wordpress]# docker build -t
wordpress:v2 .
Sending build context to Docker daemon 19.26 MB
Sending build context to Docker daemon
Step 0 : FROM orchardup/php5
Pulling repository orchardup/php5
7113324d9d9e: Download complete
511136ea3c5a: Download complete
e2aa6665d371: Download complete
f0ee64c4df74: Download complete
2209cbf9dcd3: Download complete
5e019ab7bf6d: Download complete
Status: Downloaded newer image for
orchardup/php5:latest
---> 7113324d9d9e
Step 1 : ADD . /code
---> 3286cb3866e2
Removing intermediate container f43239d824e7
Successfully built 3286cb3866e2

[root@docker01 wordpress]# docker images
REPOSITORY          TAG          IMAGE
ID                  CREATED     VIRTUAL SIZE
wordpress           v2          3286cb3866e2
    20 seconds ago    348.4 MB
orchardup/php5      latest      7113324d9d9e
    9 months ago     330.1 MB
```

```
[root@docker01 wordpress]# docker-compose up
Creating wordpress_db_1...
Pulling image orchardup/mysql:latest...
Pulling repository orchardup/mysql
061b756f7e0d: Download complete
Status: Downloaded newer image for
orchardup/mysql:latest
Creating wordpress_web_1...
Building web...
Step 0 : FROM orchardup/php5
---> 7113324d9d9e
..
db_1 | 150301 14:54:42 InnoDB: highest
supported file format is Barracuda.
db_1 | 150301 14:54:42 InnoDB: Waiting for
the background threads to start
db_1 | 150301 14:54:43 InnoDB: 5.5.38 started;
log sequence number 1595675
db_1 | 150301 14:54:43 [Note] Server hostname
(bind-address): '0.0.0.0'; port: 3306
db_1 | 150301 14:54:43 [Note] - '0.0.0.0'
resolves to '0.0.0.0';
db_1 | 150301 14:54:43 [Note] Server socket
created on IP: '0.0.0.0'.
```

Getting started with Compose and Wordpress

Install MySQL and WordPress Using Docker

- Commit new image

```
[root@docker01 wordpress]# docker commit -m "0301_status" -a "khoj@osci.kr"
6ac193d6121f wordpress_web_1_0301
8c637330a125862362018c135139791e39d043babd3ae5aaaab358f99bbf60d1
[root@docker01 wordpress]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
wordpress_web_1_0301	latest	8c637330a125	9 seconds ago	348.4 MB
wordpress_web	latest	0e8891e3f841	9 minutes ago	348.4 MB
wordpress	v2	3286cb3866e2	12 minutes ago	348.4 MB

```
[root@docker01 wordpress]# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
75303b0dd2ed	wordpress_web:latest	"php -S 0.0.0.0:8000	32 seconds ago	Up 31
seconds	0.0.0.0:8000->8000/tcp	wordpress_web_1		
04f4f1e6b3a1	orchardup/mysql:latest	"/usr/local/bin/run"	35 seconds ago	Up 34
seconds	3306/tcp	wordpress_db_1		

Nginx website

Nginx dockerfile

- Dockerfile

```
#Dockerfile

FROM ubuntu:14.04
MAINTAINER hojin kim "khoz@osci.kr"
ENV REFRESHED_AT 2015-03-09

RUN apt-get update
RUN apt-get -y -q install nginx

RUN mkdir -p /var/www/html/website
ADD nginx/global.conf /etc/nginx/conf.d/
ADD nginx/nginx.conf /etc/nginx/

EXPOSE 80
```

```
#global.conf

server {
    listen          0.0.0.0:80;
    server_name     _;

                    root      /var/www/html/website;
                    index     index.html index.htm;

    access_log     /var/log/nginx/default_access.log;
    error_log      /var/log/nginx/default_error.log;
}
```

Nginx website

Nginx dockerfile

- nginx.conf

```
#nginx.conf

user www-data;
worker_processes 4;
pid /run/nginx.pid;
daemon off;

events { }
dd
http {
    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;
    keepalive_timeout 65;
    types_hash_max_size 2048;
    include /etc/nginx/mime.types;
    default_type application/octet-stream;
    access_log /var/log/nginx/access.log;
    error_log /var/log/nginx/error.log;
    gzip on;
    gzip_disable "msie6";
    include /etc/nginx/conf.d/*.conf;
}
```

```
#index.html
<head>
<title>Test website</title>
</head>
<body>
<h1>This is a test website</h1>
</body>
```

Nginx website

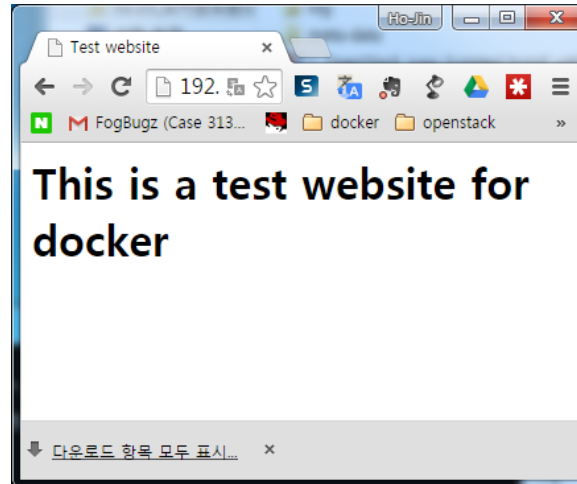
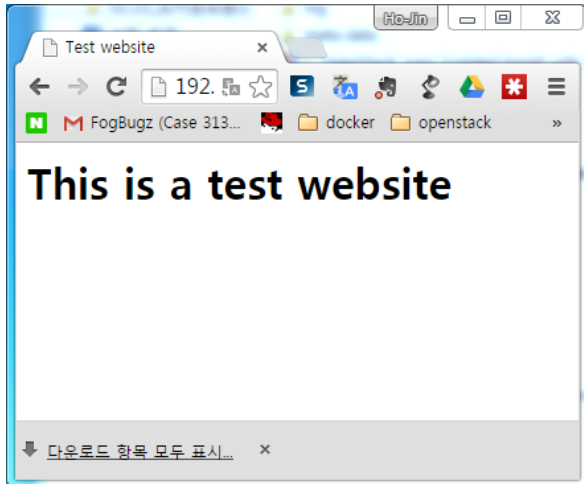
Nginx dockerfile

- nginx.conf

```
Dockerfile
nginx:
    global.conf  nginx.conf
website:
    index.html
```

```
[root@docker nginx]# docker build -t oscinfra/nginx .
```

```
[root@docker nginx]#
docker run -d -p 80 --name website -v $PWD/website:/var/www/html/website oscinfra/nginx nginx
```



Contents

1. Openstack 인프라 구축 (4 node 구성) [30분]
2. Openstack 위에 VM 생성 [20분]
3. docker 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. Docker로 WEB서비스 구축 [15분]
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. Docker로 jenkins 구현 [30분]

Openstack 위에 Docker로 WEB서비스 구축 [15분]

docker image 가지고 와서 이미지 만들기

- docker pull tutum/wordpress

```
[root@juno-compute ~]# docker pull tutum/wordpress
```

```
[root@juno-compute ~]# docker save tutum/wordpress | glance image-create --is-public=True --container-format=docker --disk-format=raw --name tutum/wordpress
```

```
+-----+
| Property      | Value                                |
+-----+
| checksum      | c343cc7afce50d264640f3238943c6de   |
| container_format | docker                               |
| created_at    | 2015-03-11T06:12:39                 |
| deleted       | False                                |
| deleted_at    | None                                  |
| disk_format   | raw                                   |
| id            | 570f59ed-a227-43b7-9be1-3ad9b85f49a7 |
| is_public     | True                                  |
| min_disk      | 0                                     |
| min_ram       | 0                                     |
| name          | tutum/wordpress                     |
| owner         | 3c402245243f443ebc2aa39605641be1   |
| protected     | False                                 |
| size          | 492773376                            |
| status        | active                                |
| updated_at    | 2015-03-11T06:14:19                 |
| virtual_size  | None                                  |
+-----+
```


Openstack 위에 Docker로 WEB서비스 구축 [15분]

docker image 가지고 와서 이미지 만들기

- glance image

```
[root@juno-compute ~]# glance image-list
```

```
[root@juno-compute ~]# glance image-list
```

ID	Name	Disk Format	Container Format	Size	Status
707b9dfc-68e4-4a9c-b7bc-708127473a56	centos7	qcow2	bare	989069312	active
43d87a3d-e1da-4eac-86d3-33a8a4a0e62d	cirros-0.3.3-x86_64	qcow2	bare	13200896	active
570f59ed-a227-43b7-9be1-3ad9b85f49a7	tutum/wordpress	raw	docker	492773376	active

```
[root@juno-compute ~]# neutron net-list
```

id	name	subnets
00f8214c-fd7a-43f6-b469-6b78492adfff	admin-net	a16ec435-0daa-4959-82c9-b6b6f50b9627 10.0.1.0/24
39e1abb0-d9bf-4f78-8cc6-88f0267e2b09	ext-net	b74b68c0-84e7-4506-9169-1a1ff72ceb6f 192.168.0.0/24
ddba5520-bc65-4762-a531-b1bfacd1b11e	test	b9bfb210-7c24-4c3c-809f-75cde2e5dd6f 10.0.2.0/24

```
[root@juno-compute ~]# nova boot --image "tutum/wordpress" --flavor m1.tiny --key-name osc --nic net-id=00f8214c-fd7a-43f6-b469-6b78492adfff WordPress
```

Openstack 위에 Docker로 WEB서비스 구축 [15분]

인스턴스 floating IP 주고, 연결하기

인스턴스

인스턴스 이름	이미지 이름	IP 주소	크기	키 패어	상태	가용성 존	작업	전원 상태	생성된 이후 시간	작업
WordPress	tutum/wordpress	10.0.1.7 192.168.0.216	m1.tiny	osc	작동중	nova	None	실행	29 minutes	스냅샷 생성
centos	centos7	10.0.1.5 192.168.0.213	m1.small	osc	정지됨	nova	None	상태 아님	2 hours,59 minutes	스냅샷 생성
test_instance3	cirros-0.3.3-x86_64	10.0.1.2 192.168.0.211	m1.tiny	osc	작동중	nova	None	상태 아님	7 hours,59 minutes	스냅샷 생성

3 항목을 보여줍니다.

```

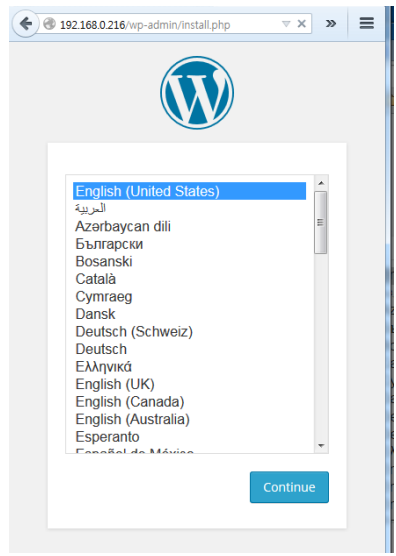
192.168.0.181/dashboard/project/instances/9a4d63d-9184-416b-aa6f-1691d1d19139/console
-> An empty or uninitialized MySQL volume is detected in /var/lib/mysql
-> Installing MySQL ...
-> Done!
-> Waiting for confirmation of MySQL service startup
-> Creating database wordpress in MySQL
-> Creating database wordpress
-> Done!
-> Creating MySQL admin user with random password
ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2)
ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2)
-> Done!

You can now connect to this MySQL Server using:

mysql -uadmin -pZf4c7fd9K0 -h<host> -P<port>

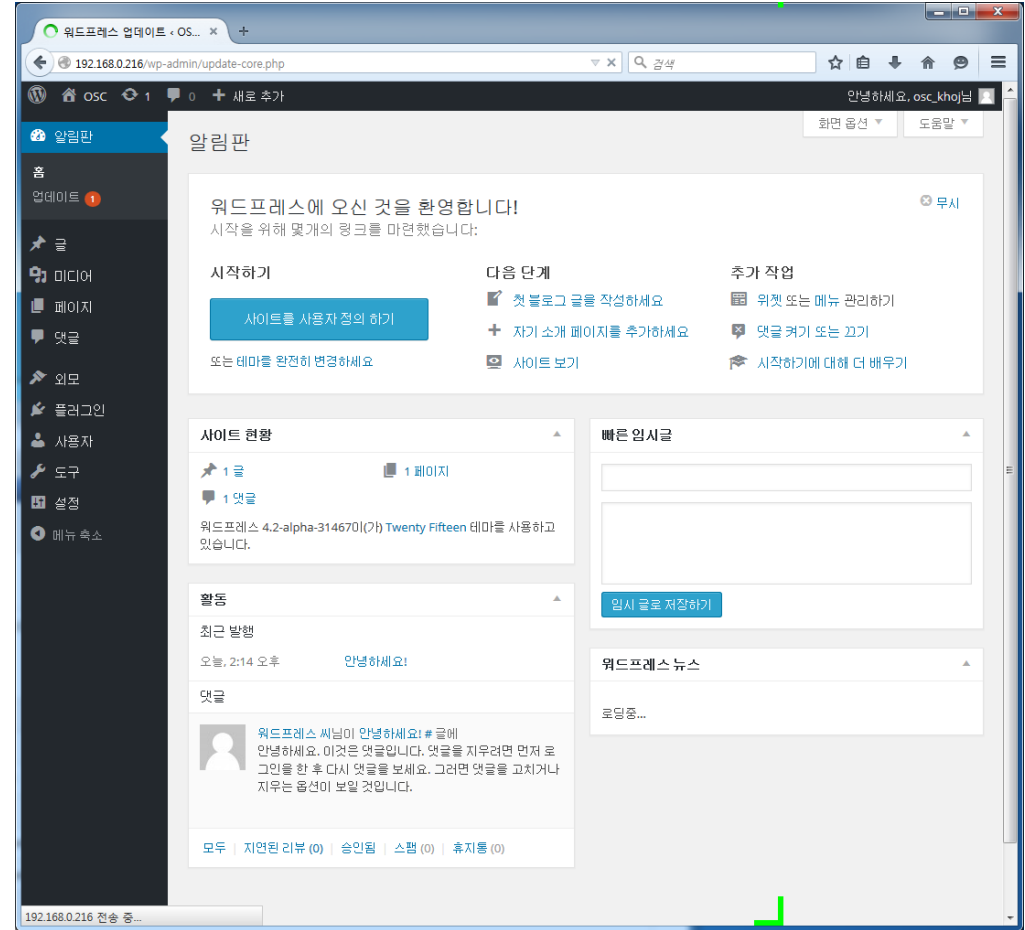
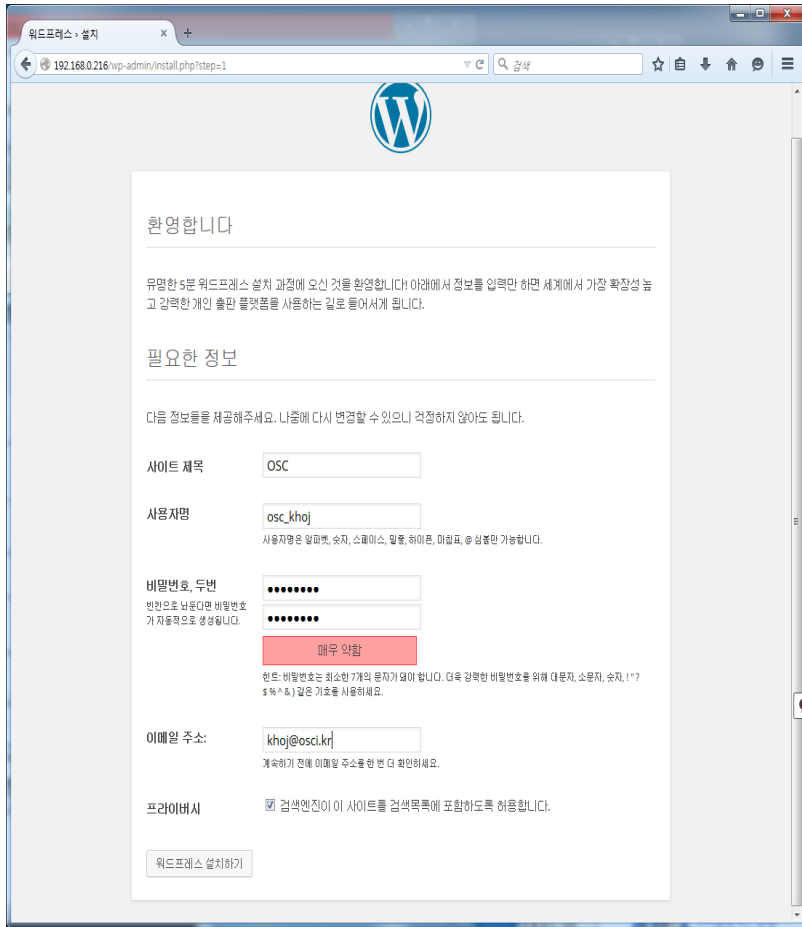
Please remember to change the above password as soon as possible!
MySQL user 'root' has no password but only allows local connections

mysqladmin: connect to server at 'localhost' failed
error: 'Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2)'
Check that mysqld is running and that the socket: '/var/run/mysqld/mysqld.sock' exists!
/usr/lib/python2.7/dist-packages/supervisor/options.py:235: UserWarning: Supervisor is running as root and it is searching for its configuration file in default locations
(including its current working directory); you probably want to specify a "-c" argument specifying an absolute path to a configuration file for improved security.
'Supervisord is running as root and it is searching '
2015-03-11 03:55:49.976 CRIT Supervisor running as root (no user in config file)
2015-03-11 03:55:49.976 WARN Includ extra file "/etc/supervisor/conf.d/supervisord-mysqld.conf" during parsing
2015-03-11 03:55:49.976 WARN Includ extra file "/etc/supervisor/conf.d/supervisord-apache2.conf" during parsing
2015-03-11 03:55:49.005 INFO RPC interface 'supervisor' initialized
2015-03-11 03:55:49.005 CRIT Server 'unix_http_server' running without any HTTP authentication checking
2015-03-11 03:55:49.006 INFO supervisord started with pid 1
2015-03-11 03:55:50.000 INFO spawned: 'mysqld' with pid 779
2015-03-11 03:55:50.010 INFO spawned: 'apache2' with pid 780
2015-03-11 03:55:51.426 INFO success: mysqld entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2015-03-11 03:55:51.438 INFO success: apache2 entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
    
```



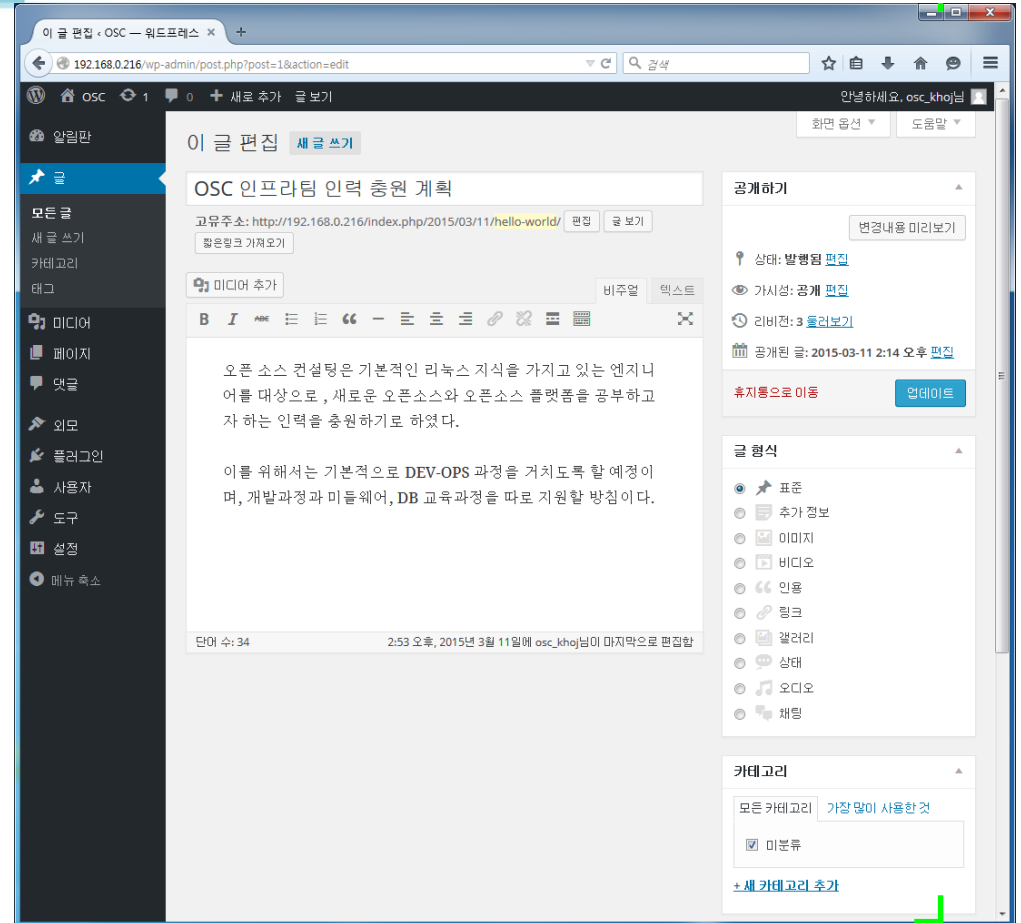
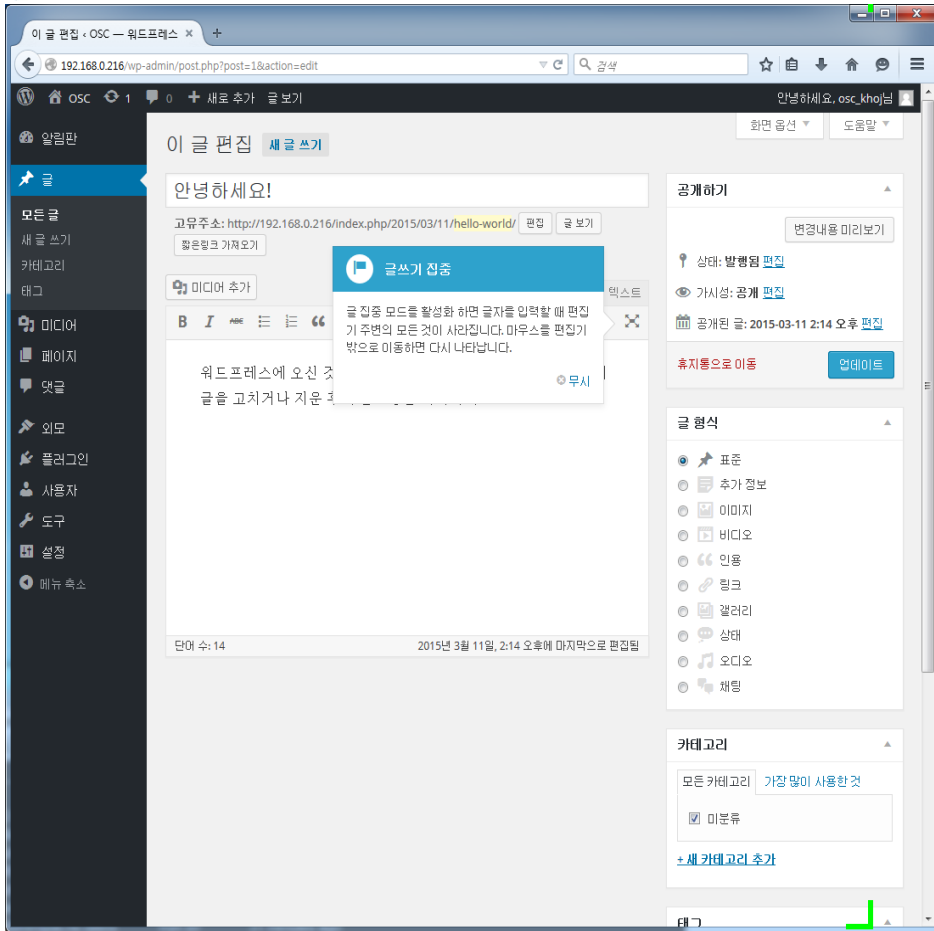
Openstack 위에 Docker로 WEB서비스 구축 [15분]

웹서비스 창



Openstack 위에 Docker로 WEB서비스 구축 [15분]

웹 서비스 창



Openstack 위에 Docker로 WEB서비스 구축 [15분]

docker image 가지고 와서 이미지 만들기

- docker pull tutum/mysql

```
[root@juno-compute ~]# docker pull tutum/mysql
```

```
[root@juno-compute ~]# docker save tutum/mysql:latest | glance image-create --is-public=True --container-format=docker --disk-format=raw --name tutum/mysql:latest
```

Property	Value
checksum	fa22fdac9cfda75cb9ecf67ed6d974c1
container_format	docker
created_at	2015-03-11T08:59:20
deleted	False
deleted_at	None
disk_format	raw
id	54c5a2f8-e1ae-475d-9eb6-0be8c38315f4
is_public	True
min_disk	0
min_ram	0
name	tutum/mysql:latest
owner	3c402245243f443ebc2aa39605641be1
protected	False
size	332313600
status	active
updated_at	2015-03-11T09:00:18
virtual_size	None

Openstack 위에 Docker로 WEB서비스 구축 [15분]

docker image 가지고 와서 이미지 만들기

- glance image

```
[root@juno-compute ~]# glance image-list
```

ID	Name	Disk Format	Container Format	Size	Status
707b9dfc-68e4-4a9c-b7bc-708127473a56	centos7	qcow2	bare	989069312	active
43d87a3d-e1da-4eac-86d3-33a8a4a0e62d	cirros-0.3.3-x86_64	qcow2	bare	13200896	active
54c5a2f8-e1ae-475d-9eb6-0be8c38315f4	tutum/mysql:latest	raw	docker	332313600	active
570f59ed-a227-43b7-9be1-3ad9b85f49a7	tutum/wordpress	raw	docker	492773376	active

```
[root@juno-compute ~]# neutron net-list
```

id	name	subnets
00f8214c-fd7a-43f6-b469-6b78492adfff	admin-net	a16ec435-0daa-4959-82c9-b6b6f50b9627 10.0.1.0/24
39e1abb0-d9bf-4f78-8cc6-88f0267e2b09	ext-net	b74b68c0-84e7-4506-9169-1a1ff72ceb6f 192.168.0.0/24
ddba5520-bc65-4762-a531-b1bfacdlb11e	test	b9bfb210-7c24-4c3c-809f-75cde2e5dd6f 10.0.2.0/24

```
[root@juno-compute ~]# nova boot --image "tutum/mysql:latest" --flavor m1.tiny --key-name osc --nic net-id=00f8214c-fd7a-43f6-b469-6b78492adfff WordPress
```

Openstack 위에 Docker로 WEB서비스 구축 [15분]

docker image 가지고 와서 이미지 만들기

- mysql contact

```
[root@juno-compute ~]# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
5ba04d7d55c9	tutum/mysql:latest	"/run.sh"	5 minutes ago	Up 5 minutes
nova-9b9de363-820a-459c-964f-ef1de66a5634	tutum/wordpress:latest	"/run.sh"	2 hours ago	Up 2 hours

```
[root@juno-compute ~]# docker logs 5ba04d7d55c9
=> An empty or uninitialized MySQL volume is detected in /var/lib/mysql
=> Installing MySQL ...
=> Done!
=> Creating admin user ...
=> Waiting for confirmation of MySQL service startup, trying 0/13 ...
=> Creating MySQL user admin with random password
=> Done!
=====
You can now connect to this MySQL Server using:

mysql -uadmin -pSXgwTukrk2fK -h<host> -P<port>
```

Openstack 위에 Docker로 WEB서비스 구축 [15분]

docker image 가지고 와서 이미지 만들기

- mysql connect

```
[root@juno-controller rootwrap.d]# mysql -uadmin -pSXgwTukrk2fK -h 192.168.0.213 -P
3306
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.5.41-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.

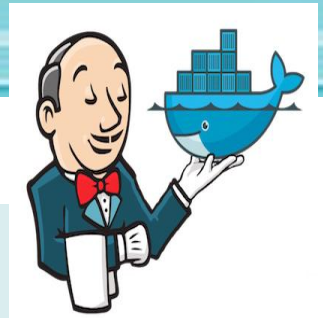
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
MySQL [(none)]> show databases;
+-----+
| Database                |
+-----+
| information_schema      |
| mysql                   |
| performance_schema     |
+-----+
3 rows in set (0.00 sec)
```


Contents

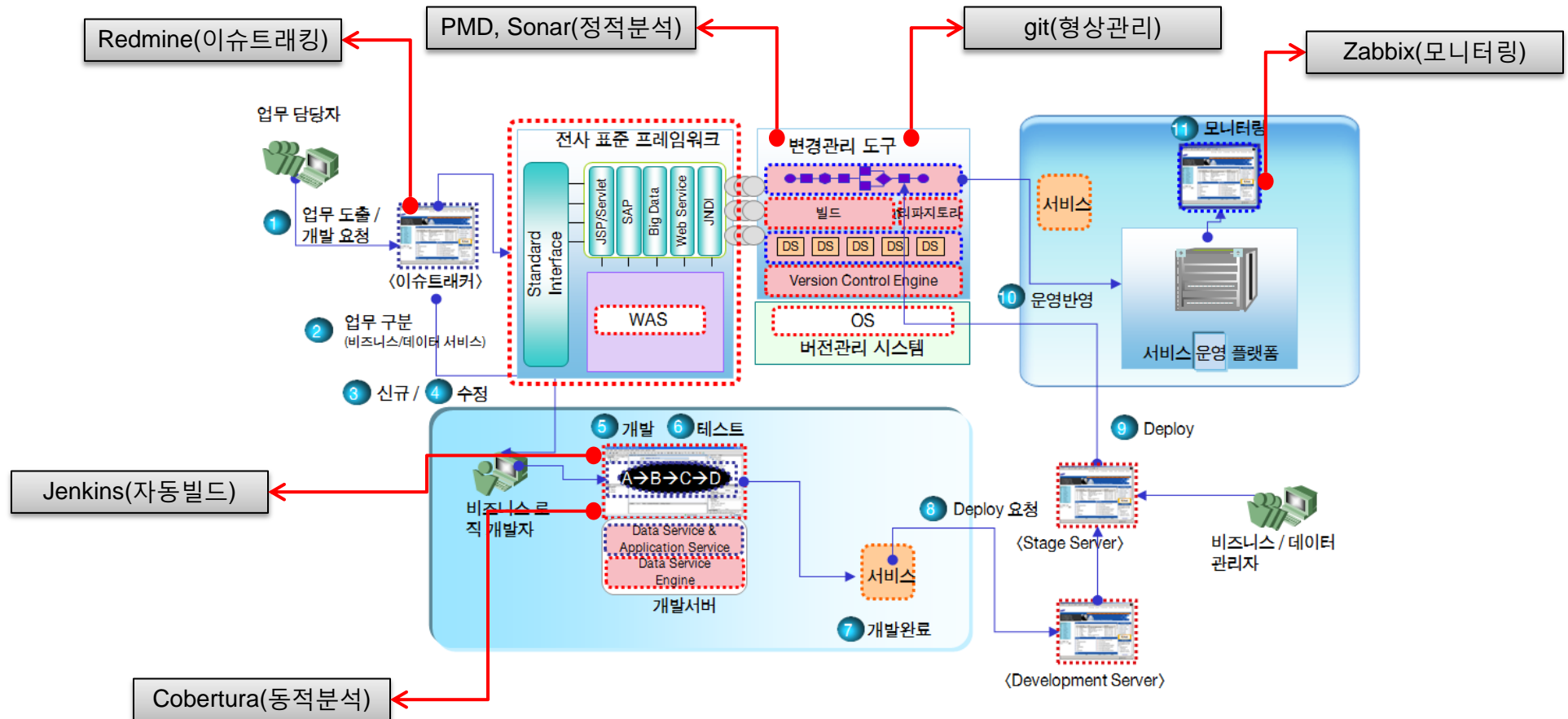
1. Openstack 인프라 구축 (4 node 구성) [30분]
2. Openstack 위에 VM 생성 [20분]
3. docker 구축 기초 [30분]
4. 오픈스택에 docker를 연결 [30분]
5. Docker로 WEB서비스 구축 [30분]
6. Openstack 위에 Docker로 WEB서비스 구축 [15분]
7. **Docker로 jenkins 구현 [30분]**

What to do during 30 min.



Docker + openstack + Jenkins

- Jenkins / 가상환경



Change openstack setting

Dockfile - ubuntu 환경에서 직접 해 봄.

```
[root@docker01 jenkins]# cat Dockerfile
FROM ubuntu:14.04
MAINTAINER james@example.com
ENV REFRESHED_AT 2014-06-01

RUN apt-get update -qq && apt-get install -qqy curl
RUN curl https://get.docker.io/gpg | apt-key add -
RUN echo deb http://get.docker.io/ubuntu docker main >
/etc/apt/sources.list.d/docker.list
RUN apt-get update -qq && apt-get install -qqy iptables ca-certificates lxc openjdk-6-
jdk git-core lxc-docker

ENV JENKINS_HOME /opt/jenkins/data
ENV JENKINS_MIRROR http://mirrors.jenkins-ci.org

RUN mkdir -p $JENKINS_HOME/plugins
RUN curl -sf -o /opt/jenkins/jenkins.war -L $JENKINS_MIRROR/war-
stable/latest/jenkins.war
RUN for plugin in chucknorris greenballs scm-api git-client git ws-cleanup ;\
do curl -sf -o $JENKINS_HOME/plugins/${plugin}.hpi \
-L $JENKINS_MIRROR/plugins/${plugin}/latest/${plugin}.hpi ; done
ADD ./dockerjenkins.sh /usr/local/bin/dockerjenkins.sh
RUN chmod +x /usr/local/bin/dockerjenkins.sh
VOLUME /var/lib/docker
EXPOSE 8080
ENTRYPOINT [ "/usr/local/bin/dockerjenkins.sh" ]
```

Change openstack setting

dockerjenkins.sh

```
root@ubuntu:~/jenkins# cat dockerjenkins.sh
#!/bin/bash

# First, make sure that cgroups are mounted
correctly.
CGROUP=/sys/fs/cgroup

[ -d $CGROUP ] ||
  mkdir $CGROUP

mountpoint -q $CGROUP ||
  mount -n -t tmpfs -o uid=0,gid=0,mode=0755
  cgroup $CGROUP || {
    echo "Could not make a tmpfs mount. Did
you use -privileged?"
    exit 1
  }

# Mount the cgroup hierarchies exactly as they
are in the parent system.
for SUBSYS in $(cut -d: -f2 /proc/1/cgroup)
do
  [ -d $CGROUP/$SUBSYS ] || mkdir
  $CGROUP/$SUBSYS
  mountpoint -q $CGROUP/$SUBSYS ||
    mount -n -t cgroup -o $SUBSYS cgroup
  $CGROUP/$SUBSYS
done
```

```
# Now, close extraneous file descriptors.
pushd /proc/self/fd
for FD in *
do
  case "$FD" in
    # Keep stdin/stdout/stderr
    [012])
      ;;
    # Nuke everything else
    *)
      eval exec "$FD>&-"
      ;;
    esac
done
popd

docker -d &
exec java -jar /opt/jenkins/jenkins.war
```

name error가 발생하여.
DEBIAN_FRONTEND=noninteractive 후
apt-get install lxc-docker 수행

Change openstack setting

Build

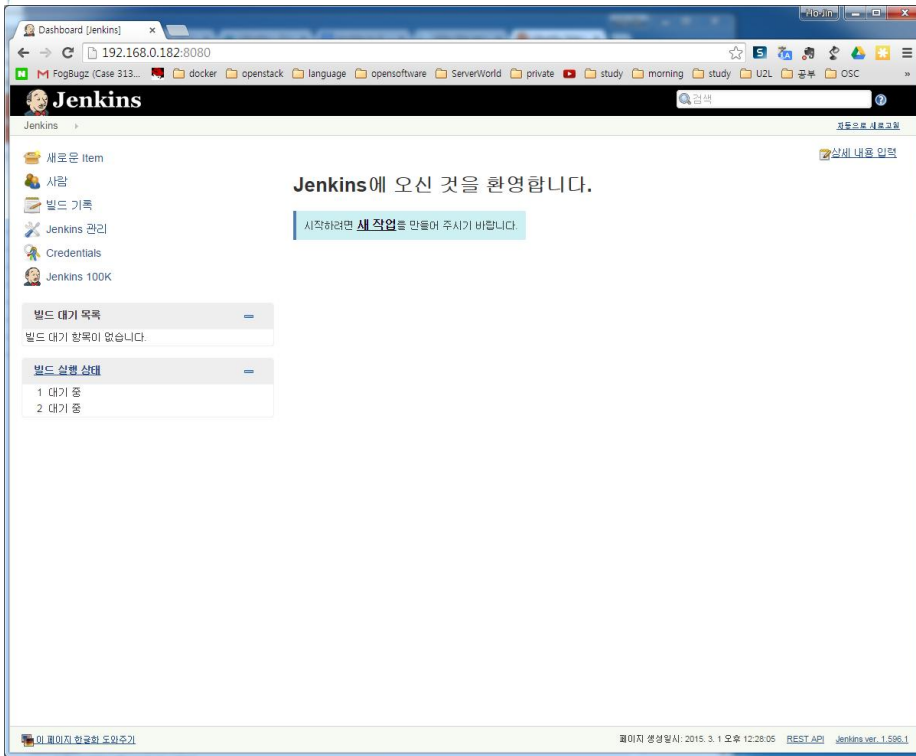
```
root@ubuntu:~/jenkins# docker build -t khoj/dockerjenkins . Sending
build context to Docker daemon 4.096 kB
Sending build context to Docker daemon
Removing intermediate container 8b902e230d08
Successfully built ab6d4f9c8830
root@ubuntu:~/jenkins# docker image
docker: 'image' is not a docker command. See 'docker --help'.
root@ubuntu:~/jenkins# docker images
REPOSITORY          TAG                IMAGE ID           CREATED            VIRTUAL SIZE
khoj/dockerjenkins  latest            ab6d4f9c8830      8 seconds ago     581.2 MB
ubuntu              14.04             2d24f826cb16      8 days ago        188.3 MB
ubuntu              latest            2d24f826cb16      8 days ago        188.3 MB

root@ubuntu:~/jenkins# docker run -p 8080:8080 --name khoj -privileged -d
khoj/dockerjenkins

Warning: '-privileged' is deprecated, it will be replaced by '--privileged' soon. See usage.
cb40bc62e8a56116e04b7c4d41f431da30ca1aeb1b4efab476c9bec6884de828
```

Change openstack setting

Build



```
root@ubuntu:~/jenkins# docker logs khoj /proc/self/fd /
```

Item 이름

Freestyle project
이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드 할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.

이름

설명

[Escaped HTML] [미리보기](#)

오래된 빌드 삭제

이 빌드는 매개변수가 있습니다

빌드 안함 (프로젝트가 다시 빌드를 할 때까지 새로운 빌드가 실행되지 않습니다.)

필요한 경우 concurrent 빌드 실행

고급 프로젝트 옵션

Quiet period

재시도 횟수

업스트림 프로젝트가 빌드하는 동안 빌드 멈춤

다운스트림 프로젝트가 빌드하는 동안 빌드 멈춤

사용자 빌드 경로 사용

디렉터리

표시 이름

Keep the build logs of dependencies

소스 코드 관리

None

CVS

CVS Projectset

Git

Repositories

Change openstack setting

Excute shell

```
# Build the image to be used for this job.
IMAGE=$(docker build . | tail -1 | awk '{ print $NF }')

# Build the directory to be mounted into Docker.
MNT="$WORKSPACE/.."

# Execute the build inside Docker.
CONTAINER=$(docker run -d -v "$MNT:/opt/project" $IMAGE /bin/bash -c 'cd
/opt/project/workspace && rake spec')

# Attach to the container so that we can see the output.
docker attach $CONTAINER

# Get its exit code as soon as the container stops.
RC=$(docker wait $CONTAINER)

# Delete the container we've just used.
docker rm $CONTAINER

# Exit with the same value as that with which the process exited.
exit $RC
```

Change openstack setting

success

 빌드 #1 (2015. 3. 1 오후 12:43:05)

5 min 34 sec 전에 업데이트 됨.
소요 [2 min 9 sec](#)

 [상세 내용 입력](#)



No changes.



익명 사용자에 의해 시작됨



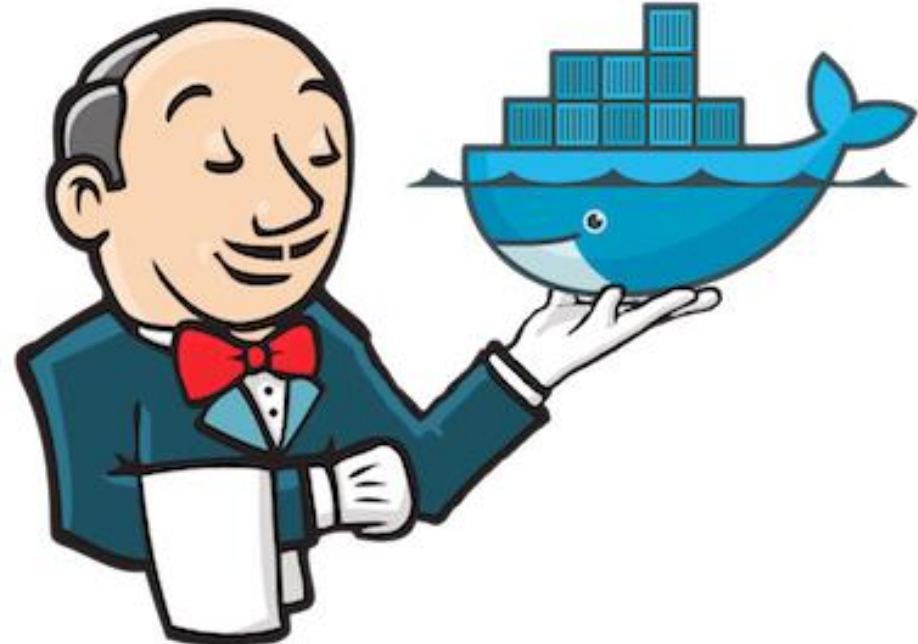
git

Revision: e4830a7c4a17b2614aee8c158ecd5da434c4412c

- refs/remotes/origin/master



[Test Result](#) (실패가 없습니다)



한국의 레드햇 꿈꾸는 오픈소스 전문가 집단 '오픈소스컨설

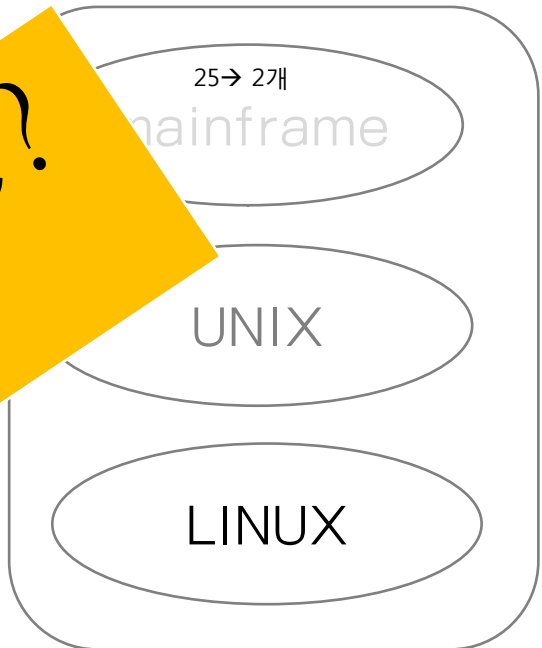


[디지털데일리 심재석기자] IT산업에서 오픈소스소프트웨어(OSS)가 차지하는 비중이 커지고 있는 가운데, 국내에서 OSS 전문가 집단을 표방하며 시장에 영향력을 넓히고 있는 회사가 있어 주목된다. 주인공은 매우 작...

- ▶ 2015년을 뜨겁게 달굴 오픈소스를 만나다
- ▶ 오픈스택, 클라우드 세상의 중심에 서다
- ▶ MS마저 사랑 고백한 오픈소스, 대세 넘어 필수로 떠올라

삼성 갤럭시S6 OLED 디스플레이 패널, 화질 평점 '최고등급'
▶ 삼성디스플레이, 투과율 30% 55인치 투명 OLED 개발...연...
▶ 애플, '애플워치' OLED 패널 조달 다변화 추진...LGD에...

예상 못한 청와대 카드...케이블...
▶ 청와대 전 수석 입장...미국...
▶ 지상파 출신 NO...케이...



Why don't you join here?
We need YOU!!!

IBM

pSeries 650
Service Guide
SA38-0612-02

Chapter 2. Diagnostics Overview
Maintenance Analysis Procedure
System LEDs
System Attention
Checkpoints
FRU Isolation
FRU Replacement
Sequencing

Accessing the Service Processor
Accessing the Service Processor
Resetting the Service Processor
Resetting the Service Processor
Resetting the Service Processor

Chapter 3. Maintenance
Entry MAP
Quick Entry MAP
Quick Entry MAP-Table
MAP 1020: Problem Determination
Purpose of This MAP
MAP 1230: Linux Platform Error
Purpose of This MAP

IBM

Contents

...activations
...Storage Pool
...placements explained
Power Integrated Facility for Linux described

Guillermo Corti
Sylvain Delabarre
Ho Jin Kim
Ondrej Plochy
Marcos Quezada
Gustavo Santos

ibm.com/redbooks

IBM

IBM PowerVC
Introduction and Configuration

OpenStack compatibility for integration with cloud software stacks
Integration of server and storage virtualization
IBM PowerVM virtualization

Bruno Blanchard
Guillermo Corti
Sylvain Delabarre
Ho Jin Kim
Ondrej Plochy
Marcos Quezada
Gustavo Santos

ibm.com/redbooks

감사합니다