관리 모듈(Nova) 설치 가이드

목차

개요

본 가이드는 오픈소스 기반 클라우드 관리 솔루션의 결과물로서, 오픈 소스 기반의 IaaS 관리 소프트웨어인 OpenStack Compute(‘Nova”)의 설치를 위한 상세한 설명을 제공 하고자 한다.

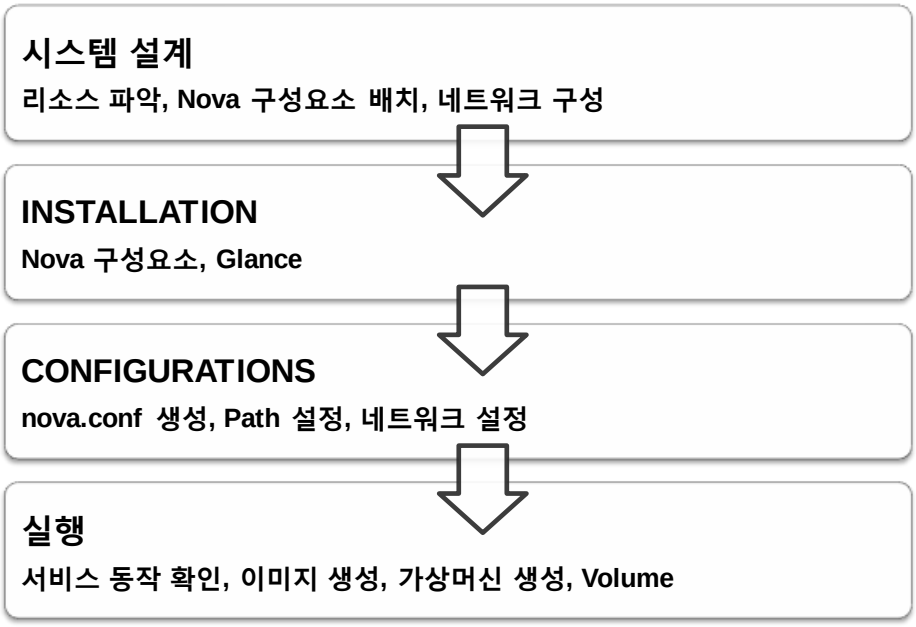
1 설치 과정

일반적인 설치를 위한 과정은 아래의 그림과 같다.

첫째, 자신이 확보할 수 있는 시스템 리소스를 파악하고 Nova의 모듈 별 특성에 맞추어 적절하게 시스템 구성을 위한 설계를 진행한다.

둘째, 앞서 설계한 시스템의 구성에 맞추어 시스템에 Nova를 설치하고, 설치된 모듈 별 configuration을 설정한다.

셋째, 각 시스템 별로 설치된 Nova 모듈을 구동시키고 적절한 설치와 구동 여부를 점검한다.



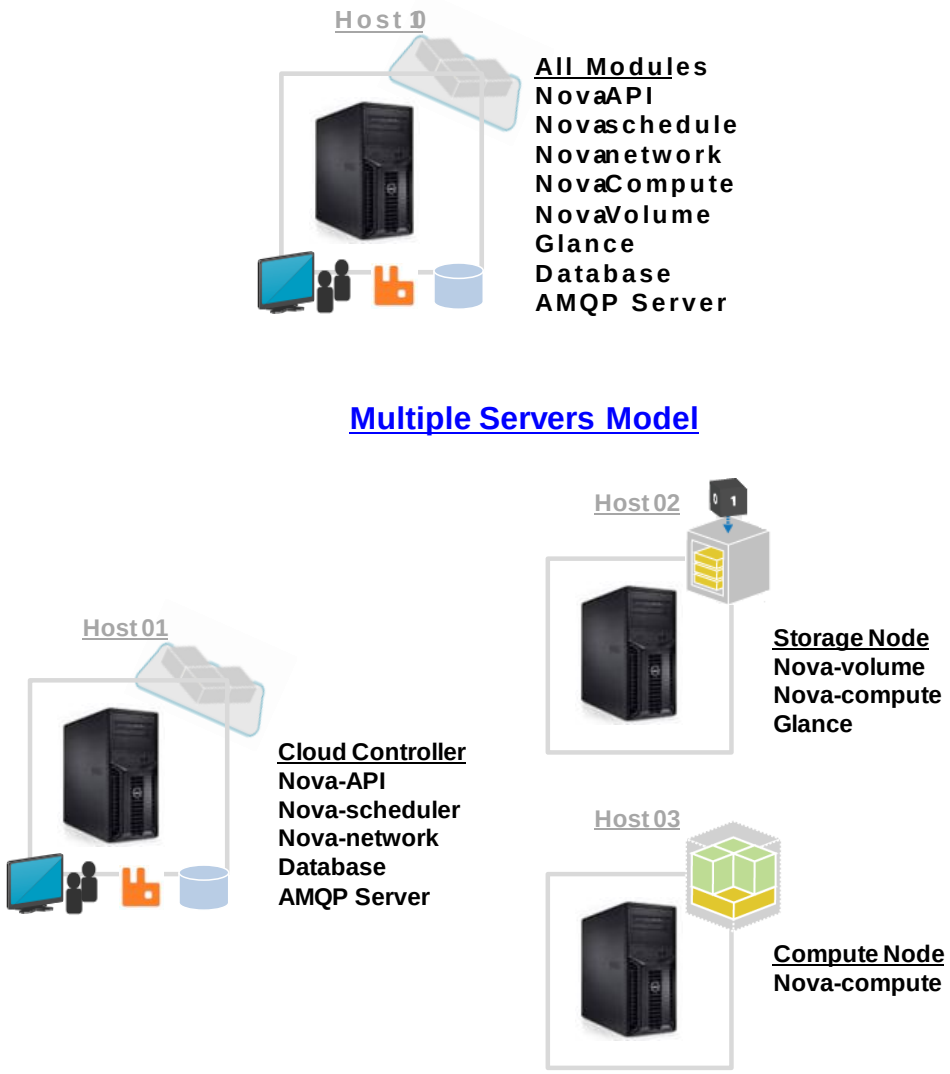
[그림 1. Nova 설치 과정]

2 시스템 설계

본 가이드에서는 Nova의 2가지 구성에 대한 시스템 설치 방법을 제공하고자 한다. 첫째, Nova 및 구동에 필요한 모듈을 모두 하나의 시스템에 설치하는 Single Server model, 둘째, Nova의 구성 요소들을 각 목적에 맞추어 여러 대의 시스템에 나누어 설치하는 Multiple Servers model이다.

Nova는 기본적으로 5개의 서비스(Compute, Scheduler, API, Network, Volume)로 구성되어 있으며, 이들 서비스와 더불어 이미지 서비스인 Glance, Database, 메시징 서비스(Rabbitmq-server)를 설계 시 함께 고려해야 한다. 이 모든 서비스들은 서로 다른 호스트에서 독립적으로 실행이 가능하며, 리소스를 제공해야 하는 compute, network와 같은 서비스는 여러 대의 서버에서 실행시키면 부하를 분산시킬 수 있다.

아래 그림은 이 가이드에서 소개할 각 Single / Multiple Servers의 Reference Model이다.



[그림 2. Reference Model]

3 Nova 설치(Installation & Configuration)

3.1 싱글 서버 모델(Single Server model)

Single Server Model은 Nova의 각 모듈 및 설치 요소들을 하나의 싱글 머신에 모두 설치하는 방식이다. Nova의 설치를 통해 5가지의 기본적인 구성요소(Compute, Scheduler, API, Network, Volume)와 함께 database와 rabbitmq등의 소프트웨어 설치가 가능해지며, 3.2과 3.3에 소개되는 glance 설치와 volume 구성을 통해 하나의 싱글 머신에 모든 요소의 설치와 구성이 완성된다.

Nova의 기본적인 설치 방법으로는 소스코드를 이용한 설치와 패키지를 이용한 설치의 2가지 방법이 있으며 그에 대한 설명은 아래와 같다.

3.1.1 Nova 코드 설치(소스코드 방식)

Bazzr 및 Launchpad 사용법 참고: http://wiki.openstack.org/LifeWithBzrAndLaunchpad

1) bazzar를 설치한다.

```
# apt-get install bzr
```

2) 아래의 bzr branch 명령을 이용하면 nova라는 디렉토리(이후 $NOVA_DIR) 안에 최신 소스가 저장된다. bzr pull을 이용하면 최신 revision으로 업데이트가 가능하다.

```
# bzr branch lp:nova
# bzr pull
```

3) Nova 뿐 아니라 Nova가 동작하는데 필요한 패키지들을 한 번에 설치하는 가장 좋은 방법은 소스코드 안에 들어있는 "nova.sh" 스크립트를 이용하는 것이다.

```
# $NOVA_DIR/contrib/nova.sh install
```

4) 'PATH' 환경 변수에 Nova 실행 파일들이 있는 경로를 추가해준다.

```
# vi ~/.bashrc
(중략)
PATH=$NOVA_DIR/bin:$PATH
# source ~/.bashrc
```

5) Nova 실행에 필요한 각종 데이터파일들을 저장할 디렉토리를 생성한다. Nova에서는 이 디렉토리를 'state_path'라고 부른다. state_path는 관리자의 편의에 맞게 설정하되 Nova 실행 파일들이 참조할 수 있도록 뒤에 설명할 configuration 파일에 명시해 주어야 한다.

이때, state_path는 "nova" 계정이 read/write 가능하도록 '755'이상으로 설정한다.

```
# mkdir -p /openstack/state
# mkdir /openstack/state/instances
# mkdir /openstack/state/networks
# chmod 755 /openstack –R
```

※ 위 예시에서는 state_path를 /openstack/state로 설정하였으나 다르게 설정해도 관계없다.

6) Nova 실행 파일들의 log를 저장할 디렉토리를 생성해 준다.

```
# mkdir /openstack/logs
# chmod 755 /openstack/logs
```

7) Nova API 서비스 설정을 저장 해 둔 api-paste.ini 파일을 /etc/nova 아래에 복사 해 둔다.

```
# cp $NOVA_DIR/etc/nova/api-paste.ini /etc/nova/
```

3.1.2 Nova 코드 설치(패키지 방식)

1) Nova repository를 등록한다. 'ppa:nova-core/trunk'를 등록하면 최신 revision을 받을 수 있다. 정식 release 버전을 원한다면 'ppa:nova-core/release'를 등록한다.

```
# apt-get install python-software-properties
# add-apt-repository ppa:nova-core/trunk
# apt-get update
```

2) Nova 패키지와 그 외 필요한 패키지들을 설치한다.

```
# apt-get install rabbitmq-server
# apt-get install python-nova
# apt-get install nova-common nova-doc nova-api nova-network nova-scheduler nova-compute euca2ools
```

3.1.3 Configuration

1) Nova의 각종 옵션들은 configuration 파일(/etc/nova/nova.conf)에 저장된다. Configuration 파일에 아래의 설정 값들을 넣어주되, 색깔로 강조된 항목은 적절하게 수정한다.

```
# mkdir /etc/nova
# vi /etc/nova/nova.conf
----------------------------------------------------------------------
--verbose
--nodaemon
--dhcpbridge_flagfile=/etc/nova/nova.conf
```

```
--public_interface=eth0
--sql_connection=mysql://root:nova@localhost/nova
--auth_driver=nova.auth.dbdriver.DbDriver
--libvirt_type=qemu
--state_path=/openstack/state
--logdir=/openstack/log
--------------------------------------------------------------------------------
```

2) 추가로 아래의 네트워크 설정 FLAG들 중 자신의 네트워크 환경에 맞는 것을 넣어준다. 이상의 FLAG들은 Nova를 실행하는데 기본적인 것들로 각 FLAG에 대한 자세한 설명 및 그 이외의 FLAG들은 링크를 참조한다.

```
[ VLAN 모드 (default) ]
--network_manager=nova.network.manager.VlanManager
--public_interface=eth0
--vlan_interface=eth1
--vlan_start=100

[ Flat 모드 ]
--network_manager=nova.network.manager.FlatManager

[ FlatDHCP 모드 ]
--network_manager=nova.network.manager.FlatDHCPManager
--flat_interface=eth0
```

nova.conf 플래그 참조:
http://docs.openstack.org/cactus/openstack-compute/admin/content/reference-for-flags-in-nova-conf.html

3.2 Glance

Glance는 Swift, S3, File System을 Object Storage로 사용할 수 있도록 해 주는 인터페이스로 Nova의 기본 이미지 서비스(FLAGS.image_service)이다.

3.2.1 Glance 설치

Glance는 package 형태로 배포되고 있으므로 쉽게 설치가 가능하다. Glance는 API와 Registry 서비스를 실행할 호스트와 compute 호스트에 설치한다. 단, 각 호스트에 설치된 Glance의 버전이 동일해야 하기 때문에 한 호스트의 Glance 버전을 update 하였다면, 다른 호스트에서도 update를 해 주어야 한다.

```
# sudo add-apt-repository ppa:glance-core/trunk
```

```
# sudo apt-get update
# sudo apt-get install glance
```

3.2.2 Glance 설정

1) Glance를 설치하면 /etc/glance 디렉토리 아래에 glance-api.conf와 glance-registory.conf 파일이 생긴다. 각 설정 항목이 이해하기 쉽게 되어있으므로 자신이 사용할 backend storage에 맞게 설정을 해 준다. S3나 Swift 사용이 어렵다면 'default_store=file'로 설정해 file system을 사용하도록 한다.
2) Log 파일이 저장 될 'log_file' 경로의 owner를 glance로 바꿔준다.

```
# chown –R glance:nogroup /var/log/glance
```

3.2.3 Glance 실행

Glance 서버 역할을 할 호스트에서 아래의 명령으로 Glance를 실행한다. 실행 후, 아래와 같이 각 서비스 별로 2개씩 프로세스가 떠야 정상적으로 동작한다.

```
# service glance-api start
# service glance-registry start
# ps –aef | grep glance
glance    1118     1  0 Jun30 ?        00:00:00 su -c glance-registry glance
glance    1119     1  0 Jun30 ?        00:00:00 su -c glance-api glance
glance    1133  1118  0 Jun30 ?        00:00:00 /usr/bin/python /usr/bin/glance-registry
glance    1134  1119  0 Jun30 ?        00:00:00 /usr/bin/python /usr/bin/glance-api
```

3.2.4 이미지 등록

Glance를 이용하여 이미지를 등록하는 방법은 간단하다. 자세한 설명은 링크를 참조한다. 여기서는 UEC 이미지를 등록하는 방법을 예제로 기술하였다.

Glance CLI 사용법: http://glance.openstack.org/glance.html

```
# glance add name='uec-kernel' is_public=true < ubuntu-10.04-uec-amd64-vmlinuz-virtual
Added new image with ID: 1
# glance add name='uec-machine' is_public=true kernel_id=1 < ubuntu-10.04-uec-amd64.img
Added new image with ID: 2
```

```
# glance index --host=glance.api.host.ip
ID                   Name                           Size
---------------- ------------------------------ --------------
2                    ubuntu-machine         1476395008
```

```
1               ubuntu-kernel           4106592
```

3.2.5 Configuration

Nova에서 Glance를 사용하기 위해서는 Glance API가 실행되고 있는 서버의 주소를 configuration 파일에 명시해 주어야 한다.

```
--glance_api_servers=serverip:9292
```

3.3 Volume

Nova-volume은 가상머신에서 사용할 수 있는 block level의 storage를 제공하는 서비스로 iSCSI방식의 Linux LVM을 지원한다. Volume은 instance사이에서 쉽게 이동이 가능하나 한번에 하나의 instance에만 attach가 가능하다.

3.3.1 Volume 구성 요소 설치

Nova의 Volume을 구성하기 위해서는 nova-volume의 lvm구성을 위한 요소들과 compute node간의 iscsi통신을 위한 요소들의 설치가 필요하다.

```
# apt-get install lvm2
# apt-get install nova-volume(앞에서 nova를 설치하였다면 생략)
# apt-get install iscsitarget(volume node쪽에 설치)
# apt-get install open-iscsi(compute node쪽에 설치)
```

3.3.2 Volume 구성

Volume구성을 위한 파티션 설정 및 nova-volumes의 이름의 volume group을 생성한다.

```
# fdisk /dev/sdb
: n (new partition) --> p (primary partition) --> 1 (partition num 1) --> enter (first cyl) --> enter (last cyl) --> : t (partition system id) --> 8e (Linux LVM) -->: w (write table)
# partprove
# pvcreate /dev/sdb1
# vgcreate nova-volumes /dev/sdb1
```

3.3.3 iSCSI 설정

1) iSCSI target 설정 및 서비스를 실행시킨다.

```
# sed -i 's/false/true/g' /etc/default/iscsitarget
# service iscsitarget start(volume node쪽)
# service open-iscsi start(compute node쪽)
```

3.3.4 Configuration 및 실행

Nova.conf파일에 iscsi discovery를 위한 prefix를 설정해주고 volume모듈을 실행한다.

```
# --iscsi_ip_prefix=192.168.(multiple node사용시 nova-volume 서버의 full ip로 설정)
# service nova-volume start
```

3.4 멀티 서버 모델(Multiple Servers mode)

Multiple Servers Model은 Nova의 각 구성 요소들을 목적에 맞추어 여러 대의 시스템에 나누어 설치하는 방법이다. 앞서 소개한 "2장 시스템 설계"의 [그림2]의 multiple servers model을 예로 설명해보면, 각 host1,2,3에 3.1에서 소개한 Nova의 설치를 동일한 방법으로 각 서버에 설치한다. 더불어 host2에 3.2와 3.3에 소개된 glance 설치 및 volume구성 작업을 진행한다. 즉, 자신이 설계하는 방법에 따라 각각의 nova의 구성요소들을 간단하게 설치함으로써, 멀티 서버 모델의 구성이 간단하게 가능해지며, 추가적으로 앞에서 기술한 설치 및 설정에 더해 각 서버의 역할에 따라 아래의 설정을 더해 줌으로써 구성이 완성된다.

3.4.1 DataBase 설정

1) 원격으로 DB 접근이 가능하도록 mysql 설정을 아래와 같이 변경해 준다.

```
# mysql –pnova
> use mysql;
> UPDATE user SET host='%' WHERE host='localhost' and user='root';
> quit;
```

2) '/etc/mysql/my.cnf'를 열어 'bind-address=127.0.0.1'이 있으면 주석 처리한다.

3) mysql-server를 재시작 한 후, 원격으로 접근이 되는지 확인해 본다.

```
# mysql -h mysql.server.ip.addr -uroot -pnova
```

3.4.2 FLAG 설정

Nova의 각 서비스들이 여러 대의 호스트에서 실행된다 하더라도 Database, 메시징 서비스(rabbitmq-server), API 서비스는 zone 내에서 공유된다. 따라서, 각 호스트의 configuration 파일에 이런 서비스들이 실행되고 있는 호스트를 명시해 주어야 한다.

```
--sql_connection=mysql://root:nova@192.168.0.8/nova
--rabbit host=192.168.0.8
--ec2_host=123.123.123.123
--ec2_dmz_host=192.168.0.8
```

```
--osapi_host=121.166.195.8/192.168.0.8
--glance_api_servers=192.168.0.4:9292
```

3.5 Nova 실행

3.5.1 DB 생성 및 네트워크 설정

Nova 실행을 위해서는 추가적으로 DB생성 및 네트워크 설정(Vlan mode)이 필요하다.

1) Nova DB를 생성하는 과정에서 UI설치시 한글 처리를 고려한다면 UTF-8로 지정하고 Table 생성한다.

```
# mysql -pnova -e 'CREATE DATABASE nova default character set utf8;'
```

2) UTF-8지정시 한 테이블에 Index를 타는 column의 바이트 수가 1000바이트를 넘게 되면 오류가 발생한다. 이를 해결하기 위해 Nova 에서 제공하는 소스를 수정해야 한다. 영향을 받는 테이블은 projects 테이블, users 테이블, user_projuect_association 테이블, user_project_role_association 테이블, user_role_association 테이블 등 5개의 테이블이다. 여기서 user_id, project_id, role 과 관련된 필드가 255 바이트로 설정된 것을 100 바이트 정도로 수정하여야 한다.

```
nova/db/sqlalchemy/migrate_repo/versions/001_austin.py

projects 테이블 관련
id 255-->100 (254 라인부근)

users 테이블 관련
id 255-->100 (371 라인부근)

user_project_association 테이블 관련
user_id 255-->100 (394 라인부근)

user_project_role_association 테이블 관련
user_id 255-->100 (414 라인부근)
project_id 255-->100 (414 라인부근)
role 255-->100 (424 라인부근)

user_role_association 테이블 관련
user_id 255-->100 (440 라인부근)
role 255-->100 (446 라인부근)
```

```
nova/db/sqlalchemy/models.py

Class User
id 255-->100

Class Project
id 255-->100

UserProjectRoleAssociation
user_id 255-->100
project_id 255-->100
role 255-->100

UserRoleAssociation
user_id 255-->100
role 255-->100

UserProjectAssociation
user_id 255-->100
project_id 255-->100
```

3) 수정이 되었다면 DB sync와 더불어 네트워크를 생성한다.

```
# nova-manage db sync
# nova-manage network create private 10.0.0.0/8 10 64
```

3.5.2 Nova 서비스 실행

위의 설정이 완료되면 이제 Nova의 각 서비스들을 실행되면 된다. 각 서비스들이 한 대의 호스트에서 모두 실행되어도 되고 여러 대의 호스트에서 나누어 실행되거나 중복 실행되어도 상관없다.

- nova-scheduler
- nova-api
- nova-compute
- nova-network
- nova-volume
- Glance API, Registry

- DataBase (MySql, Sqlite, or Postgres)
- Rabbitmq-Server

위 서비스들이 해당 서버에서 모두 실행된 후 'nova-manage service list' 명령을 실행하면 각 서비스들이 잘 동작하고 있는지 확인할 수 있다.

```
# nova-compute
# nova-api; nova-scheduler; nova-network;
# nova-manage service list
```

Binary	Host	Zone	Status	State	Updated_At
nova-compute	hostA	nova	disabled	:-)	2011-10-19 07:39:05
nova-compute	hostB	nova	disabled	:-)	2011-10-19 07:39:00
nova-scheduler	hostC	nova	enabled	:-)	2011-10-19 07:39:10
nova-network	hostD	nova	enabled	:-)	2011-10-19 07:39:10
nova-volume	hostE	nova	enabled	:-)	2011-10-19 07:39:18

3.5.3 계정 및 인증 생성

1) Nova 실행 후에는 사용자 계정 생성 및 인증키 생성이 필요하다.

```
# nova-manage user admin cloudadm
# nova-manage project create cloudprj cloudadm
# nova-manage project zipfile cloudprj cloudadm
```

```
export EC2_ACCESS_KEY="cloudadm:cloudprj"
export EC2_SECRET_KEY="fb92898a-55e9-48a4-b9dc-095518b9a35c"
export EC2_URL=http://192.168.0.8:8773/services/Cloud
export NOVA_API_KEY="cloudadm"
export NOVA_USERNAME="cloudadm"
export NOVA_PROJECT_ID="cloudprj"
export NOVA_URL="http://192.168.0.8:8774/v1.1/"
```

2) Keypair를 생성한다.

```
# . novarc
# euca-add-keypair cloudadm > cloudadm.pem
```

4 부록

4.1 VPN 사용하기

Nova가 지원하는 3개의 네트워크 모드 중 VLAN 모드를 사용한다면, 외부에서 가상머신에 접근하기 위해 VPN을 거쳐야 한다. Nova에서 VPN은 'cloudpipe'로 불리며, VPN을 실행하면 OpenVPN 서버가 자동으로 실행되는 cloudpipe 가상머신이 하나 생성되어 VLAN과 외부 네트워크를 연결해주는 식으로 동작한다.

4.1.1 Cloudpipe 이미지 생성

Cloudpipe는 OpenVPN과 OpenVPN 실행에 필요한 유틸리티들(OpenSSL, bridge-utils, unzip)이 설치된 일반적인 리눅스 이미지이면 된다.

1) UEC 이미지를 다운로드하여 압축을 푼다.

```
# wget http://uec-images.ubuntu.com/releases/10.04/release/ubuntu-10.04-amd64.tar.gz
# tar xvf ubuntu-10.04-amd64.tar.gz
# ls -l
lucid-server-uec-amd64.img
lucid-server-uec-amd64-vmlinuz-virtual
lucid-server-uec-amd64-loader
README.files
```

2) UEC 이미지를 loop device에 마운트하고 root file system을 UEC로 변경한다. 그 전에 apt가 제대로 동작할 수 있도록 resolve.conf와 apt source list를 UEC에 복사해 둔다.

```
# mkdir /mnt/ubuntu
# mount -o loop lucid-server-uec-amd64.img /mnt/ubuntu
# cp /etc/resolve.conf /mnt/ubuntu/etc/
# cp /etc/apt/source.list /mnt/ubuntu/etc/apt/
# chroot /mnt/ubuntu
```

3) OpenVPN과 필요 패키지들을 설치하고 다시 원래의 root file system으로 돌아온다.

```
# apt-get update
# apt-get install openssl openvpn bridge-utils unzip
# exit
```

4) Cloudpipe 인스턴스 부팅 시, OpenVPN을 자동실행 하는데 필요한 스크립트와 설정 파일들을 UEC 이미지에 복사해 둔다. 이 파일들은 $NOVA_DIR/doc/source/devref/ 아래에 포

함되어 있다.

```
# cp $NOVA_DIR/doc/source/devref/rc.local /mnt/ubuntu/etc/
# cp $NOVA_DIR/doc/source/devref/interfaces /mnt/ubuntu/etc/network/
# cp $NOVA_DIR/doc/source/devref/up.sh /mnt/ubuntu/etc/openvpn/
# cp $NOVA_DIR/doc/source/devref/down.sh /mnt/ubuntu/etc/openvpn/
# cp $NOVA_DIR/doc/source/devref/server.conf.template /mnt/ubuntu/etc/openvpn/
# umount /mnt/ubuntu
```

5) 이미지 준비는 끝났다. 이제 이미지를 Glance를 이용해 등록한다.

```
# glance add name='cloudpipe-kernel' is_public=true < ubuntu-10.04-uec-amd64-vmlinuz-virtual
Added new image with ID: 1
# glance add name='cloudpipe-machine' is_public=true kernel_id=1 < ubuntu-10.04-uec-amd64.img
Added new image with ID: 2
```

4.1.2 FLAG 설정

VPN을 사용하기 위해서 아래의 FLAG들을 configuration 파일에 추가한다. vpn_image_id는 Glance를 이용해 등록한 cloudpipe machine 이미지의 ID이다.

```
--vpn_image_id=2
--use_project_ca=True
--cnt_vpn_clients=5
```

4.1.3 VPN 사용법

1) VPN을 사용하는 방법은 간단하다. --project에는 VPN 인스턴스를 띄우고자 하는 프로젝트 이름을, --user에는 해당 프로젝트의 admin 사용자의 계정을 넣는다.

```
# nova-manage vpn run --project=PROJECT --user=ADMIN
# nova-manage vpn list
project              ip:port                 private_ip              state
NOVAADMPRJ           121.166.195.54:1000     10.0.0.2 cloud02 i-00000003 running up
```

2) VPN에 연결하기 위해 nova-vpn.conf 파일을 생성한다.

```
# nova-manage project zipfile PROJECT USER
# unzip nova.zip
-rw------- 1    root root 2083 2011-09-02 14:28 cacert.pem
-rw------- 1    root root 2539 2011-09-02 14:28 cert.pem
-rw------- 1    root root 1132 2011-09-02 14:28 novarc
-rw------- 1    root root 1169 2011-09-02 14:28 nova-vpn.conf
```

```
-rw-r--r-- 1    root root 8298 2011-09-02 14:28 nova.zip
-rw------- 1    root root    887 2011-09-02 14:28 pk.pem
```

3) OpenVPN Client 프로그램과 nova-vpn.conf를 이용해 VPN에 접속한다. nova-vpn.conf를 Windows나 다른 개인 PC로 옮겨 접속하는 것도 가능하다.

4.2 Windows Image 만들기

4.2.1 준비물

Windows 이미지를 만들기 위해서는 아래의 준비가 필요하다.

- Windows Install Image – ISO 파일
- 리눅스가 깔린 컴퓨터 (가상머신은 안됨)
- CMOS setup에서 CPU Virtualization를 Enabled로 설정
- virtio 드라이버파일
  http://alt.fedoraproject.org/pub/alt/virtio-win/latest/images/bin/virtio-win-1.1.16.iso

4.2.2 Machine 이미지 만들기

1) Windows를 설치할 이미지를 생성한다.

```
# kvm-img create –f qcow2 windows.img 10G
```

2) 1에서 만든 이미지에 Windows를 설치한다.

```
# kvm -m 1024 –cdrom windows.iso –drive file=windows.img,boot=on
```

3) Virtio block device 드라이버를 설치하기 위해 이미지를 한 개 더 만든다.

```
# kvm-img create –f qcow2 virtio.qcow2 1G
```

4) Virtio를 위해 만든 이미지를 연결하고 드라이버 설치파일을 연결해서 Windows 이미지를 부팅한다.

```
# kvm –m 1024 –cdrom virtio-win-1.1.16.iso –drive file=windows.img,boot=on –drive
file=virtio.qcow2,if=virtio
```

5) 부팅 후 Windows의 장치관리자에서 virtio 드라이버를 설치해 준다.

4.2.3 Ramdisk 이미지 만들기

1) Boot 이미지로 쓸 플로피 디스크 파일을 하나 만든다.

```
# ddb s=512 count=2880 if=/dev/zero of=win-boot.img
```

2) Windows 설치 이미지를 시디롬에 설정해 주고 위에서 만든 플로피 디스크를 플로피 드라이브에 설정해 Windows를 부팅한다.

```
# kvm –m 1024 –cdrom windows.iso –drive file=windows.img, boot=on –fda win-boot.img
```

3) 부팅 후, 플로피디스크를 포맷하고 CD-ROM\i386\ 아래의 ntldr, ntdetect.com 파일을 플로피디스크로 복사한다.
4) C:\windows\system32\drivers 아래의 sym_hi.sys 파일을 플로피디스크로 복사하고 이름을 Ntbootdd.sys로 변경한다.
5) 플로피디스크로 부팅해 보고 정상적으로 부팅이 되면 성공이다.

```
# kvm –m 1024 –boot a –cdrom windows.iso –drive file=windows.img,boot-on –fda win-boot.img
```

4.2.4 Kernel 이미지 만들기

1) syslinux 패키지를 설치한다.

```
# apt-get install syslinux
```

2) /usr/lib/syslinux/memdisk 파일이 Kernel 이미지로 쓰일 파일이다.

4.2.5 이미지 등록

Machine(windows.img), Ramdisk(win-boot.img), Kenel(memdisk)가 모두 준비되었다면 Glance를 이용하여 이미지를 등록하기만 하면 된다. 참고로 libvirt_type으로 KVM을 사용한다면 machine 이미지만 등록해도 동작하는데 문제가 없다.

```
# glance add name='windows-kernel' is_public=true < memdisk
Added new image with ID: 1
# glance add name='windows-ramdisk' is_public=true < win-boot.img
Added new image with ID: 2
# glance add name='windows-machine' is_public=true –kernel_id=1 –ramdisk_id=2 > win.img
Added new image with ID:3
```