

Openstack, KVM 기반

온프레미스 서비스 구축

추지나

목차

I 프로젝트 소개

1. 프로젝트명
2. 개요
3. 목적
4. 구축방법

II 프로젝트 기술

1. Private Cloud
2. NMS/SNMP

III 프로젝트 구축(상세설명)

1. 물리적 네트워크 망 구축
2. Openstack 구축
3. 프라이빗 클라우드 서비스 망 구축(Openstack)
4. 프라이빗 클라우드 서비스 망 구축(KVM)
5. NMS/SNMP로 관제구축(CACTI, PRTG, JFFNMS)

1. 프로젝트소개

프로젝트명

- 온프레미스 Private Cloud 서비스 구축 (오픈스택, KVM 기반)

개요

- 대표적인 무료 오픈소스 클라우드 소프트웨어인 오픈스택(Openstack)과 KVM을 이용하여 소규모의 프라이빗 클라우드(Private Cloud) 서비스 망을 구축하고 운영할 수 있도록 한다.
- 연습용 프라이빗 클라우드 서비스 망을 설계하고 이를 바탕으로 Openstack의 서비스들인 Keystone, Nova, Glance, Horizon, Cinder, Neutron을 사용하여 여러 VM을 구축한다.
- NMS/SMS 시스템을 통해 관제하여 직접 VM을 운영할 수 있는 기량을 습득한다.

목적

- 오픈스택 기반의 프라이빗 클라우드 서비스 망 설계
- 오픈스택 기반의 프라이빗 클라우드 서비스 망 구축
- KVM 기반의 프라이빗 클라우드 서비스 망 구축
- 프라이빗 클라우드 서비스 운영 시스템 구축 (NMS/SMS 시스템 포함)

구축방법

- 1단계 : 물리적 네트워크 망을 설계 및 구축
- 2단계 : 오픈스택 구축 (멀티노드)
- 3단계 : 오픈스택 기반의 프라이빗 클라우드 서비스 망 구축
- 4단계 : Xubuntu기반의 KVM을 통해 프라이빗 클라우드 서비스 망 구축
- 5단계 : NMS기반의 프라이빗 클라우드 서비스 운영체제 구축

2. 프로젝트 기술

1. Private Cloud란?

- NIST에 따르면 프라이빗 클라우드(Private Cloud)는 복수의 소비자로 구성된 단일 조직에 의해 독점적으로 사용되도록 프로비저닝되는 클라우드 인프라로, 이 인프라는 해당 조직, 서드파티 또는 이 둘의 조합이 소유, 관리 및 운영할 수 있으며 회사 내부 또는 외부에 위치할 수 있는 클라우드 인프라이다.
- 주문형 셀프 서비스로 사용자의 개별 관리화면을 통해 서비스를 이용할 수 있으며, 다양한 디바이스를 통해서 서비스를 접속할 수 있고, 사업자의 컴퓨팅 리소스를 여러 사용자가 공유하는 형태로 이용한다. 또한 필요에 따라 스케일업과 스케일다운이 가능한 편이라 신속한 확장성을 가지고 있다.
- 현재 필자가 이용할 기술로는 Openstack과 KVM으로 IaaS(Infrastructure as a Service)에 속한다. 이는 CPU, 하드웨어 등 컴퓨팅 리소스를 네트워크를 통해 서비스로 제공하는 모델로, 서버 가상화나 데스크톱 가상화, 온라인 스토리지 등이 바로 이의 대표적인 서비스이다.

2. 프로젝트 기술

2. NMS/SMNP 란?

- NMS(Network Management System)은 네트워크 관리 시스템으로 컴퓨터 네트워크 또는 네트워크들을 모니터링하고 관리하는데 사용되는 하드웨어와 소프트웨어의 조합을 총칭한다. 네트워크 구성요소를 관리하며 디바이스 관리 또한 담당하는데 디바이스 상태 모니터링, 시스템 성능에 영향을 미치는 상태 확인 경고 등등의 여러 솔루션을 제공가능하다. NMS에서는 이러한 작업을 수행하기 위해 다양한 프로토콜을 사용하는데 필자는 SNMP 프로토콜을 이용하여 네트워크 계층의 디바이스로부터 정보를 수집하였다.
- SNMP(Simple Network ManageMent Protocol)은 단순망관리프로토콜로 네트워크 장비 요소 간에 네트워크 관리 및 전송을 위한 프로토콜이다. UDP/IP 상에서 동작하는 단순한 형태의 메시지 교환형 네트워크 관리 프로토콜로 구현이 용이하다는 장점이 있다. 특정 정보 변수에 대한 단순한 요청/응답(Get/Set) 매커니즘으로 이루어져 있어 정보 지향적이다.

1. 물리적 네트워크 망 구축

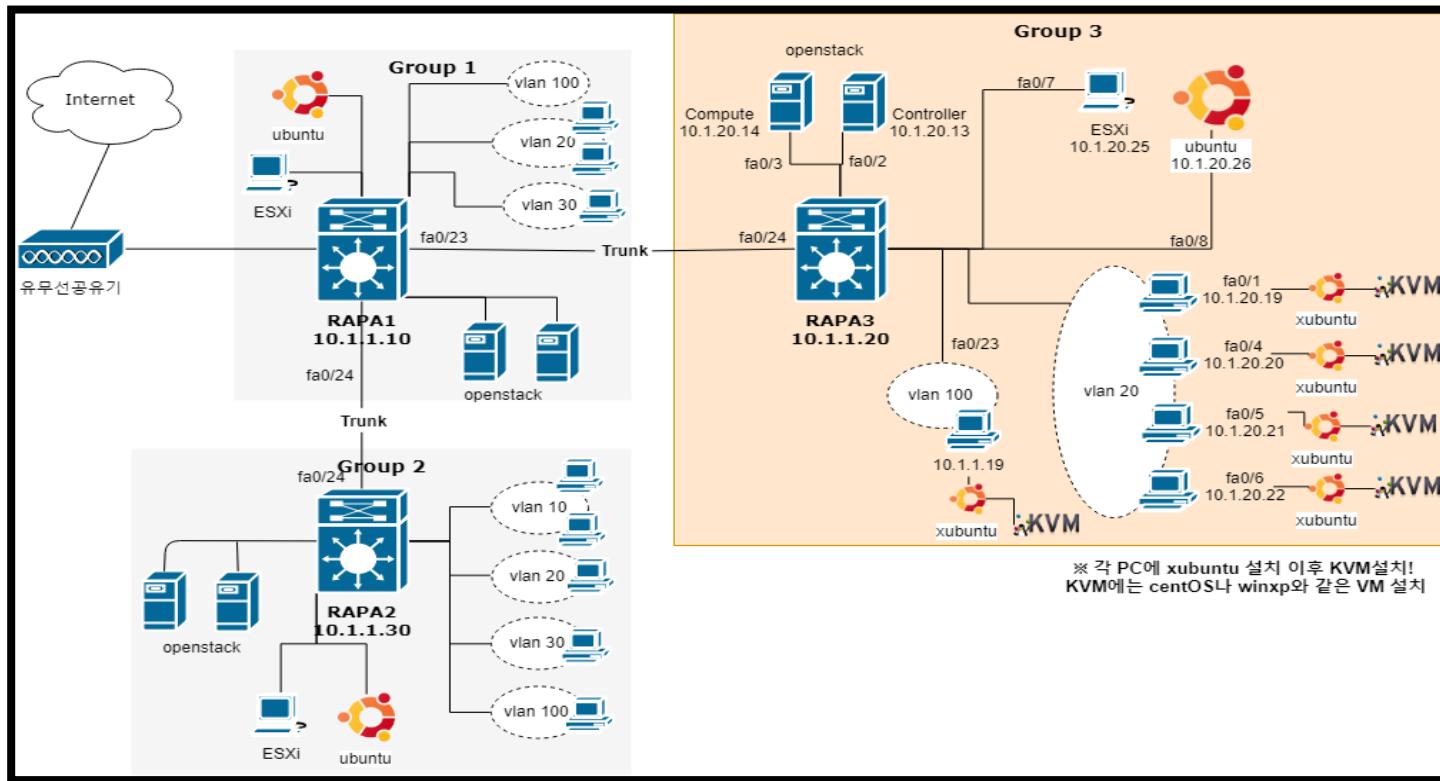
상세설명

1. 물리적인 네트워크 망 설계 및 구축

1) 물리적인 네트워크 망 설계

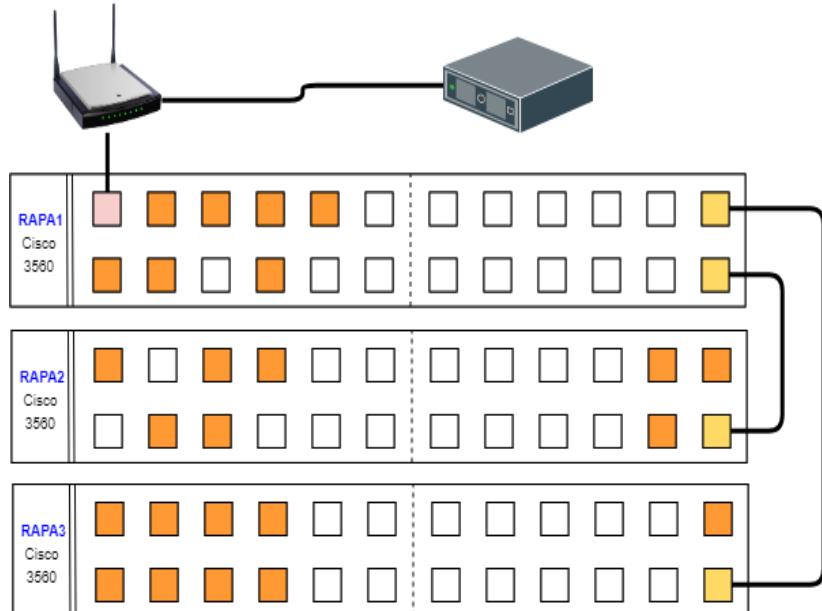
- 네트워크 구조 설계 (백본 스위치 기반)

- Catalyst 3560 스위치를 백본(코어 스위치)으로 사용하여 네트워크를 구성함
- Catalyst 3560 스위치에 추가로 Catalyst 3560 스위치를 연결하여 서버 팜을 구성함
- 백본 스위치에 일반 사용자를 위한 Catalyst 3560 스위치를 연결하여 Access망을 구성함.



- 네트워크 논리 설계 (IP Address 및 VLAN 구성)

- 서버 팝의 IP Address와 VLAN을 부여함 → 10.1.1.X, 10.1.2.X 사용, VLAN 10, VLAN 20 부여
- 일반 사용자를 위한 IP Address와 VLAN 부여함 → IP Address는 10.1.10.X, 10.1.20.X 사용
- 외부망 연동을 위한 IP Address와 VLAN 부여함 → IP Address는 192.168.7.150, VLAN은 VLAN 1000 사용
- 디폴트 게이트웨이는 IP Address 192.168.7.77을 사용함.
- 백본 스위치는 RAPA1으로 RAPA2/RAPA3 스위치를 trunk로 연결하였다.**



장비명	VLAN 주소(번호)	IP 주소
RAPA1	10.1.20.1(20)	10.1.20.10
	10.1.20.1(20)	10.1.20.20
	10.1.1.10(100)	10.1.1.100
RAPA2	10.1.10.1(10)	10.1.10.13
	10.1.20.1(20)	10.1.20.66
	10.1.30.1(30)	10.1.30.56
	10.1.1.10(100)	10.1.1.41
RAPA3	10.1.20.1(20)	10.1.20.19
	10.1.20.1(20)	10.1.20.20
	10.1.20.1(20)	10.1.20.21
	10.1.20.1(20)	10.1.20.22
	10.1.1.10(100)	10.1.1.19

스위치 설정

RAPA1 스위치



RAPA2 스위치



RAPA3 스위치



1. RAPA1 스위치 설정

스위치 명	포트 번호	vlan	trunk	subnet
RAPA1	fast ethernet 0/1-3	10		10.1.10.2/28
	fast ethernet 0/4-6	20		10.1.20.1/24
	fast ethernet 0/7-9	30		10.1.30.1/24
	fast ethernet 0/10-12	40		10.1.10.17/28
	fast ethernet 0/13-15	50		10.1.10.33/28
	fast ethernet 0/16-18	60		10.1.10.49/28
	fast ethernet 0/19	1000		192.168.7.150
	fast ethernet 0/20-22	100		10.1.1.10/24
	fast ethernet 0/23-24		O	

2. RAPA2 스위치 설정

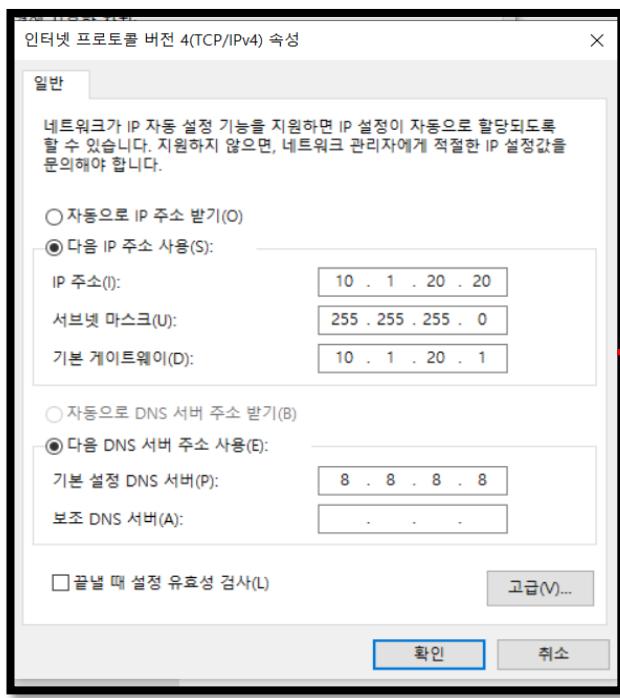
스위치 명	포트 번호	vlan	trunk	subnet
RAPA2	fast ethernet 0/1-3	10		10.1.10.1/24
	fast ethernet 0/4-6, 0/21-22	20		10.1.20.1/24
	fast ethernet 0/7-9	30		10.1.30.1/24
	fast ethernet 0/23	100		10.1.1.10/24
	fast ethernet 0/24		O	

3. RAPA3 스위치 설정

스위치 명	포트 번호	vlan	trunk	subnet
RAPA3	fast ethernet 0/1-10	20		10.1.20.1/24
	fast ethernet 0/11-14	10		10.1.10.1/24
	fast ethernet 0/23	100		10.1.1.10/24
	fast ethernet 0/24		O	

RAPA3 스위치 설정

RAPA3 스위치



```
telnet 10.1.1.20
interface Vlan1
no ip address

interface Vlan10
no ip address

interface Vlan40
no ip address

interface Vlan100
ip address 10.1.1.20 255.255.255.0
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.1.10
ip http server
ip http secure-server

line con 0
line vty 0 4
password cisco
login
line vty 5 15
login
end
```

1. 스위치 최초 설정 _ 포맷 작업

```
flash_init  
dir flash:  
boot
```

2. 호스트 네임 및 패스워드 설정

```
enable password cisco  
exit  
write
```

```
hostname RAPA3
```

3. Management Vlan 생성 _ 관리자

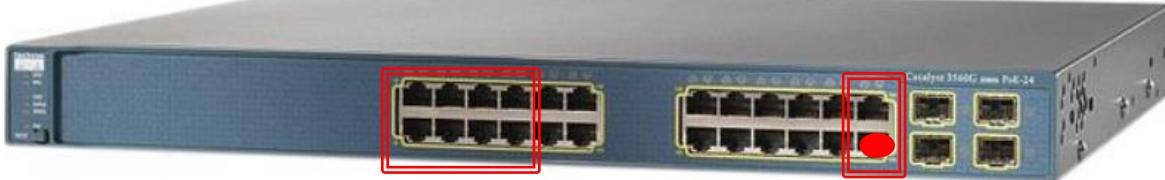
```
vlan 100
name management_vlan
interface vlan 100
ip address 10.1.1.20 255.255.255.0
no shutdown
```

```
interface fastethernet0/23
switchport mode access
switchport access vlan 100
no shutdown
```

```
line vty 0 4
password cisco
Login
```

RAPA3 스위치 설정

RAPA3 스위치



```
Telnet 10.1.1.20
spanning-tree extend system-id
no spanning-tree vlan 10-50
vlan internal allocation policy ascending
vlan20이 만들어져있음.
interface FastEthernet0/1
switchport access vlan 20
switchport mode access
interface FastEthernet0/2
switchport access vlan 20
switchport mode access
interface FastEthernet0/3
switchport access vlan 20
switchport mode access
interface FastEthernet0/4
switchport access vlan 20
switchport mode access
interface FastEthernet0/5
switchport access vlan 20
switchport mode access
interface FastEthernet0/6
switchport access vlan 20
--More--
```

```
Telnet 10.1.1.20
interface FastEthernet0/15
switchport mode access
interface FastEthernet0/16
interface FastEthernet0/17
interface FastEthernet0/18
interface FastEthernet0/19
interface FastEthernet0/20
interface FastEthernet0/21
interface FastEthernet0/22
interface FastEthernet0/23
switchport access vlan 100
switchport mode access
interface FastEthernet0/24
switchport trunk encapsulation dot1q
switchport mode trunk
interface GigabitEthernet0/1
interface GigabitEthernet0/2
```

4. Vlan 설정

```
vlan 20
interface vlan 20
ip address 10.1.20.1 255.255.255.0
no shutdown
```

```
int range fa0/1-10
switchport mode access
switchport access vlan 20
no shutdown
```

5. Trunk 설정

```
interface fa0/24
switchport trunk encapsulation dot1q
switchport mode trunk
```

모든 작업 이후에는 꼭 write를 해줘야
스위치에 저장가능.

(더 자세한 내용은 동봉된 워드파일 참조)

RAPA3 스위치 모니터링

* WireShark를 이용하여 monitor 설정

- 오픈스택과 통신할 모든 PC들의 트래픽을 확인하고 싶다면 모니터 PC를 설정하고 WireShark를 통해 한 번에 확인하는 방법이 있다.

➤ 현 필자의 스위치인 RAPA3을 기준으로 한 대의 컴퓨터만 모니터 PC로 연결하여 진행하였다.

1. Switch에서 모니터 PC 설정하기

```
RAPA3(config)#monitor session 1 source interface [오픈스택 포트 (ex.fastEthernet 0/2)] both
```

```
RAPA3(config)#monitor session 1 destination interface [모니터PC 포트 (ex.fastEthernet 0/1)]
```

성공 시, telnet에서 show run 했을 때,
옆과 같이 뜬다!

```
| monitor session 1 source interface Fa0/1  
| end
```

2. 모니터 PC가 인터넷 연결을 하고 싶을 때,

```
RAPA3(config)#monitor session 1 destination interface fa0/1 ingress [vlan(ex. vlan20)]
```

(가능하면 이 방법은 안 쓰는 것이 좋다.)

3. 모니터 설정 해제하기

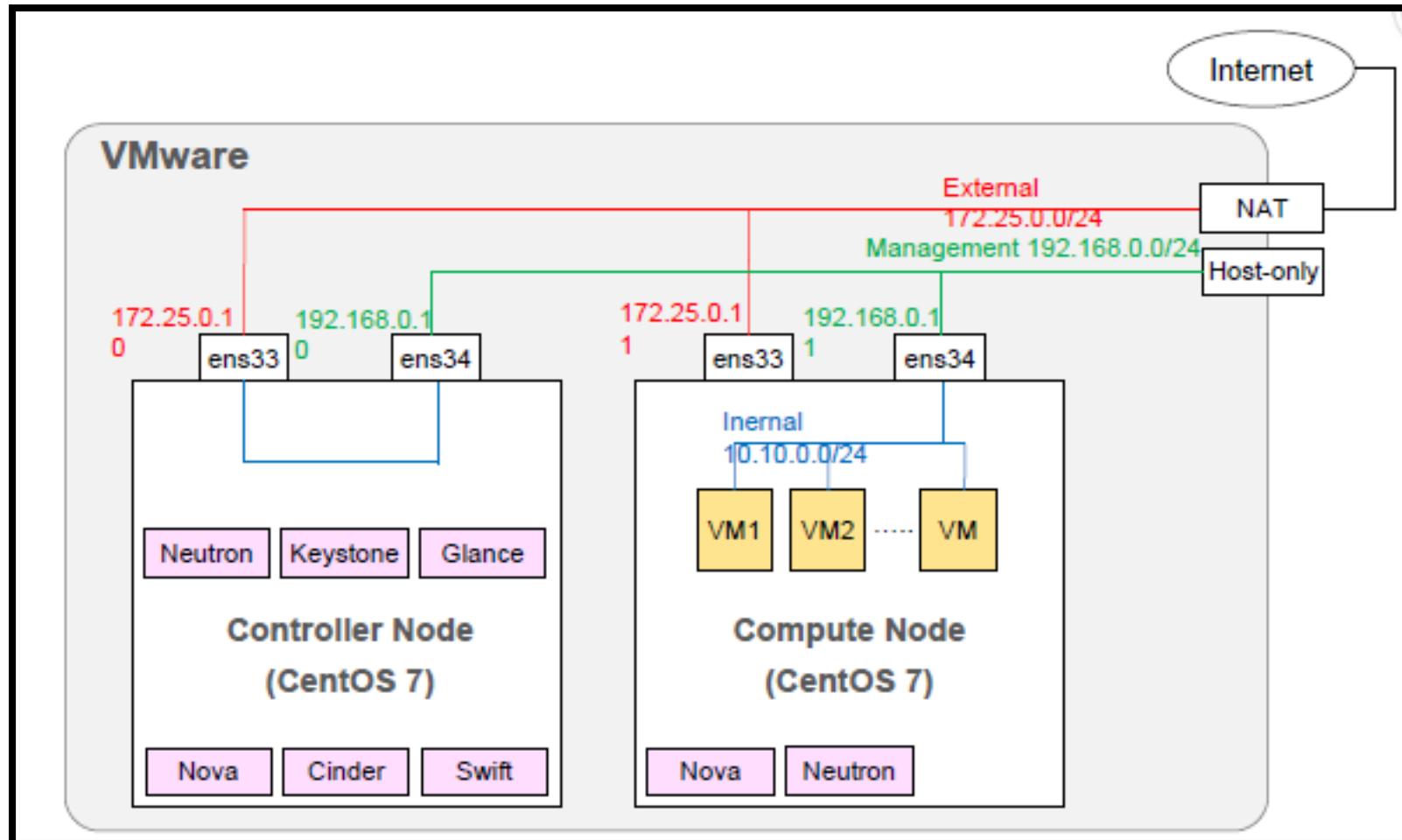
```
RAPA3(config)#no monitor session 1
```

2. Openstack 구축

2. 오픈스택 구축

1) 오픈스택 구성

- 오픈스택 Controller / Compute Node 구성도 (Multi Node 구성)



오픈스택 하드웨어 구성

- 오픈스택 Controller/Compute 노드 서버 사양

- CPU : i7 (Dual Core, INTEL-VT 지원)
- Memory : 8Gbyte
- SSD : 1Tbyte
- NIC : 1Gbps
- OS : CentOS 7.3 minimal

오픈스택 소프트웨어 구성

- 오픈스택 기반 OS : CentOS 7.3 minimal
- 오픈스택 소프트웨어 : Rocky
- 오픈스택 서비스 리스트

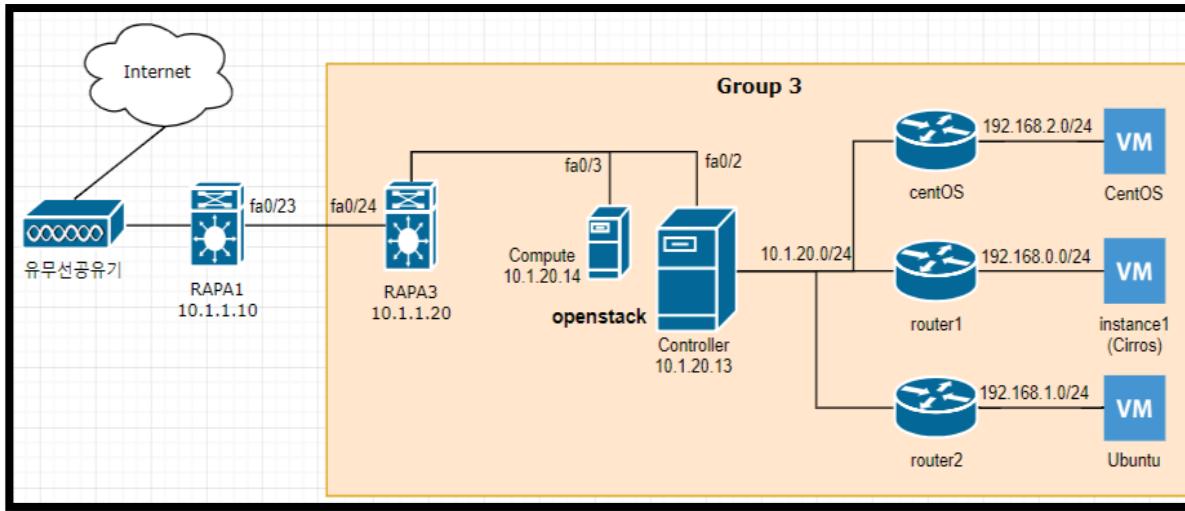
ID	Name	Type
08daf3f712364b558eaaa1bebb005f06	nova	compute
0ffb1f29c4a94e33a76bf40a8f6b4a44	glance	image
2204b93537d64dfba60ce6fdc57116ca	swift	object-store
3cfc73bdd78c48a1baa3489dfbe05f9a	cinder	volume
679a0bf34d3a4f6e994a07d50afdf8ba	placement	placement
cea7bcc869e4d53adf7c45d66297e13	neutron	network
d5aee6220a0d4168ad41472804c1fe0c	cinderv2	volumev2
f6b5d062d95441b3999231af9d1a01f9	keystone	identity
ff808f15a1a845bf95eed961ba4a936e	cinderv3	volumev3

[우리가 주로 사용할 서비스]

서비스	코드 이름	설명
컴퓨트 서비스	Nova	가상머신 관리
이미지 서비스	Glance	가상 이미지 관리
블록 스토리지	Cinder	가상머신을 위한 스토리지 관리
네트워크 서비스	Neutron	가상 네트워크 관리
인증 서비스	Keystone	사용자관리

오픈스택 전체 구성도

- 오픈스택 Controller/Compute Node 구성도



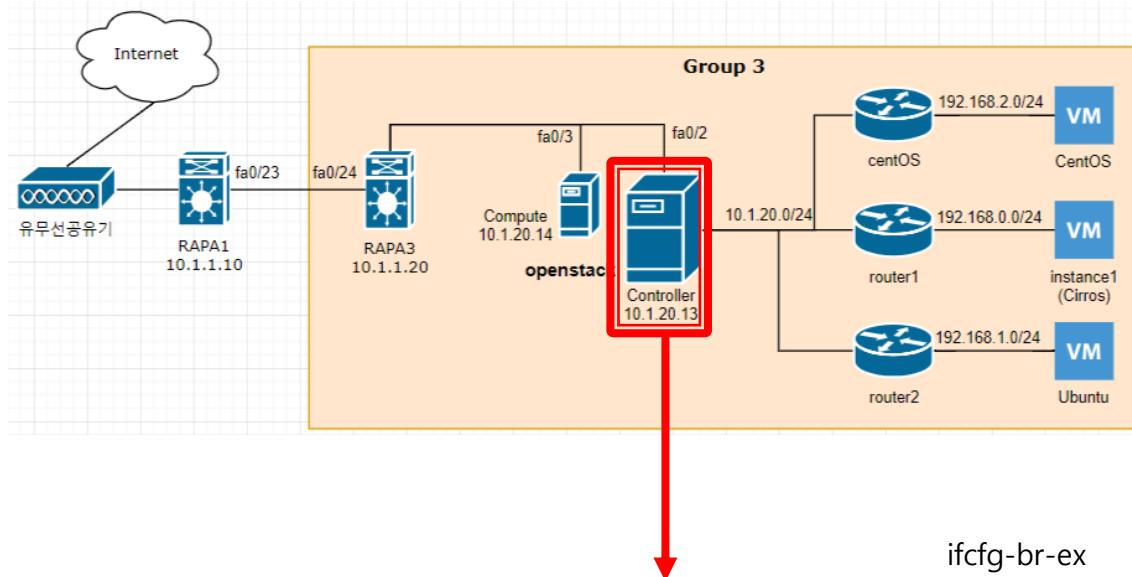
1. 구성도 설명

**CentOS기반으로 Openstack 설치진행

1. Controller / Compute Node로 설정
 - Controller IP : 10.1.20.13
 - Compute IP : 10.1.20.14
2. 프로젝트와 사용자는 각각 3개씩 설정한 후 네트워크를 설정한다.

프로젝트명	사용자명	서브넷	게이트웨이	인스턴스명	VM IP
admin	admin	192.168.2.0/24	192.168.2.1	CentOS	192.168.2.3
project1	user1	192.168.0.1/24	192.168.0.1	instance1	192.168.0.9
project2	user2	192.168.1.0/24	192.168.1.1	Ubuntu	192.168.1.8

오픈스택 설정(Controller Node)



```
ONBOOT=yes  
IPADDR=10.1.20.13  
NETMASK=255.255.255.0  
GATEWAY=10.1.20.1  
DEVICE=br-ex  
NAME=br-ex  
DEVICETYPE=ovs  
OVSBOOTPROTO=static  
TYPE=OVSBridge  
OVS_EXTRA="set bridge br-ex fail_mode=standalone"
```

- 만약 Compute node와 연결시, br-ex에 들어가있음.

Controller node 설정

```
setenforce 0  
vi /etc/selinux/config  
SELINUX=disabled  
systemctl stop NetworkManager  
systemctl disable NetworkManager
```

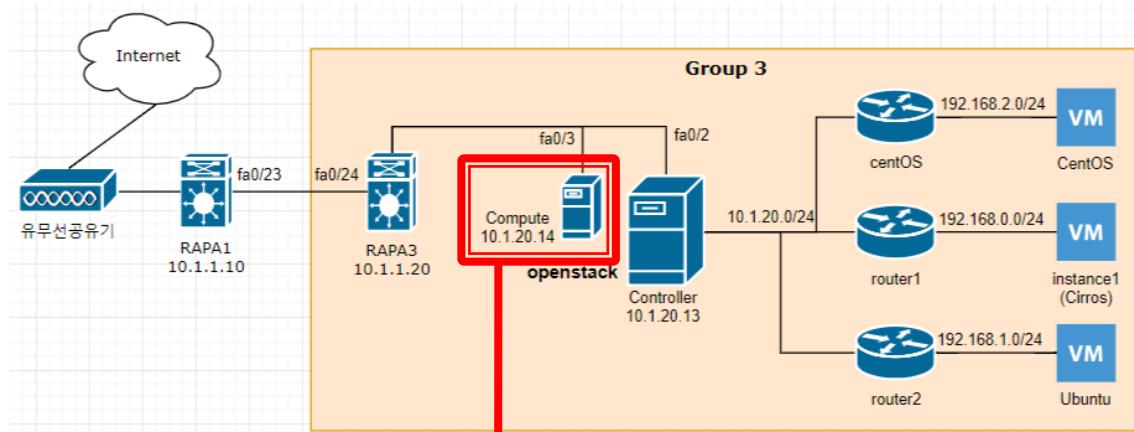
```
systemctl stop firewalld  
systemctl disable firewalld
```

네트워크 설정

```
vi /etc/sysconfig/network-scripts/ifcfg-enp5s0  
  
TYPE=Ethernet  
BOOTPROTO=static  
NAME= enp5s0  
DEVICE= enp5s0  
ONBOOT=yes  
IPADDR=10.1.20.13  
NETMASK=255.255.255.0  
GATEWAY=10.1.20.1  
DNS1=8.8.8.8
```

```
systemctl restart network
```

오픈스택 설정(Compute Node)



ifcfg-enp5s0

```
TYPE=Ethernet
BOOTPROTO=static
NAME=enp5s0
DEVICE=enp5s0
ONBOOT=yes
IPADDR=10.1.20.14
NETMASK=255.255.255.0
```

Compute와 Controller IP만 다를 뿐 나머지 설정은 거의 똑같다.

Compute Node 설정

```
setenforce 0
vi /etc/selinux/config
    SELINUX=disabled
systemctl stop NetworkManager
systemctl disable NetworkManager

systemctl stop firewalld
systemctl disable firewalld
```

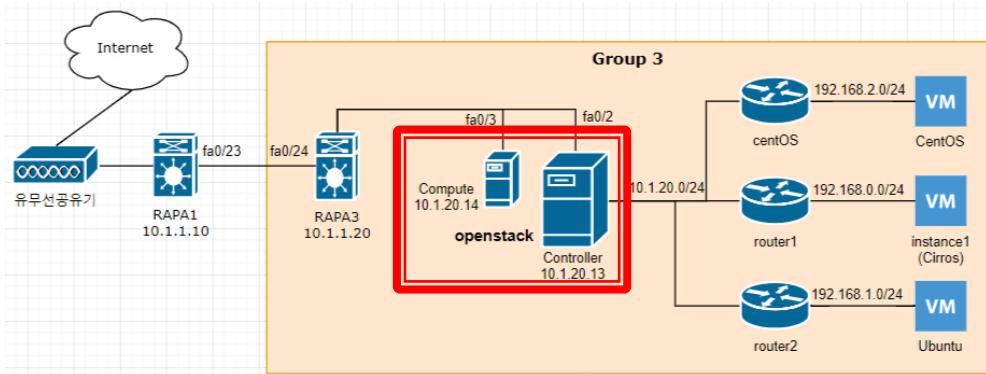
네트워크 설정

```
vi /etc/sysconfig/network-scripts/ifcfg-enp5s0

TYPE=Ethernet
BOOTPROTO=static
NAME= enp5s0
DEVICE= enp5s0
ONBOOT=yes
IPADDR=10.1.20.14
NETMASK=255.255.255.0
GATEWAY=10.1.20.1
DNS1=8.8.8.8
```

systemctl restart network

오픈스택 설치(Controller Node)



여기서 잠깐! 꼭 점검해야 되는 조건들!

- 1) SELinux disable로 되어있는지.
- 2) Firewall disable 되어있는지.
- 3) NetworkManager disable 되어있는지.
- 4) Hosts 설정 되어있는지(Controller하나만)
- 5) Hostname을 각각 설정해줬는지.
- 6) External 네트워크 NIC 설정 되어있는지
- 7) DNS 동작을 하는지.

15, 16p에 해당 조건명령어가 나와있음

Openstack 설치_Controller

```
Yum install -y centos-release-openstack-rocky  
Yum update -y  
yum install -y openstack-packstack  
packstack --gen-answer-file /root/answers.txt
```

vi /root/answers.txt

※ 위의 명령어로 answer.txt 들어가서 아래 내용 확인 및 변경

```
CONFIG_DEFAULT_PASSWORD=openstack  
CONFIG_CEILOMETER_INSTALL=n  
CONFIG_AODH_INSTALL=n  
CONFIG_NTP_SERVERS=kr.pool.ntp.org  
CONFIG_CONTROLLER_HOST=10.1.20.13  
CONFIG_COMPUTE_HOSTS=10.1.20.13,10.1.20.14  
CONFIG_KEYSTONE_ADMIN_PW=openstack  
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=extent:br-ex  
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=br-ex:enp5s0  
CONFIG_PROVISION_DEMO=n
```

packstack --answer-file /root/answers.txt

**Controller / Compute에서 설치내용 모두 확인 가능

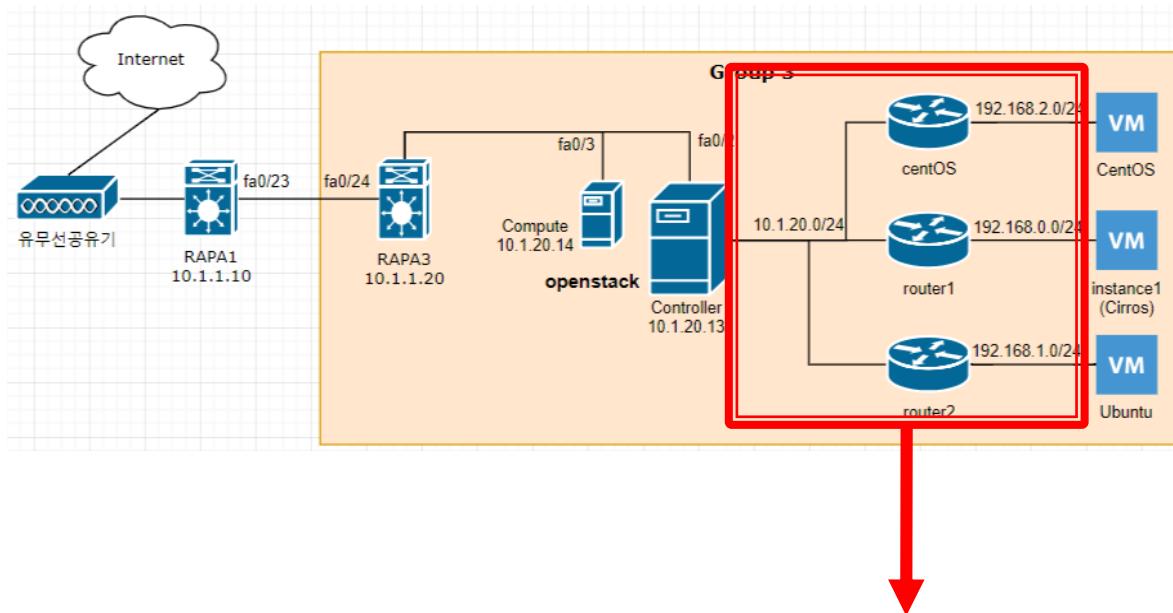
yum install openstack-utils -y

3. 프라이빗 클라우드 서비스 망 구축

- Openstack기반

오픈스택 서비스 사용하기

- Keystone / Neutron



Keystone

1. 사용자 계정 및 프로젝트 생성

사용자 계정 : admin, user1, user2

프로젝트 : admin, project1, project2

Neutron

2. 외부/내부 네트워크 설정

- 1) admin 계정에서 외부 네트워크 설정
- 2) 사용자 계정에서 내부 네트워크를 설정

외부/내부	네트워크 이름	프로젝트 이름	라우터 이름	서브넷	게이트웨이	POOL
외부	external1			10.1.20.0/24	10.1.20.1	10.1.20.50,10.1.20.100
내부	internal0	admin	CentOS	192.168.2.0/24	192.168.2.1	
	internal1	project1	router1	192.168.0.0/24	192.168.0.1	
	internal2	project2	router2	192.168.1.0/24	192.168.1.1	

Keystone 서비스 구현

Keystone 이란?

- 중앙에서 사용자에 대한 인증 디렉토리를 제공하는데 이 디렉토리는 사용자가 접근 가능한 오픈스택 서비스와 매핑된 정보를 저장하고 있다.
여러 형태의 인증 방법을 지원하는데 필자는 그 중 전통적인 방식의 사용자 ID와 패스워드, 토큰 기반 인증 시스템 방식으로 만들어보았다.

Ex)

```
unset OS_SERVICE_TOKEN
export OS_USERNAME=gaegae
export OS_PASSWORD='gaegae'
export OS_AUTH_URL=http://10.1.10.8:5000/v3
export PS1='[\u@h \W(keystone_gaegae)]\$ '
export OS_PROJECT_NAME=gaegae
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_IDENTITY_API_VERSION=3
```

Admin 계정으로 접속하여 사용자 계정인 gaegae
계정에 keystone 토큰을 부여함

gaegae 계정으로 접속하여 만든 인스턴스를 확인함

Controller에서 만든 Keystone 토큰 확인 가능

```
[root@controller ~]# ls
anaconda-ks.cfg          keystonerc_admin
answers.txt                keystonerc_user1
CentOS-7-x86_64-DVD-1810.iso  keystonerc_user2
cirros-0.3.4-x86_64-disk.img  ubuntu-14.04.3-desktop-amd64.iso
```

```
[root@controller ~]# nova list
+-----+-----+-----+-----+
| ID   | Name  | Status | Task State | Power S |
| state | Networks |           |             |
+-----+-----+-----+-----+
| f688cbe6-f7f2-4a74-8264-11e5807d51ff | CentOS7 | ACTIVE | -      | Running
| 68.20.24, 10.1.20.110 |                   |
| 79b-6d25dde819a1 | Cirros | ACTIVE | -      | Running
| 68.20.9 |           |
+-----+-----+-----+-----+
```

프로젝트 생성(Keystone)

1) 프로젝트 생성 명령어

- 프로젝트 생성 : openstack project create --domain default --description "Project1" project1

➤ Description은 넣고 싶은 설명이며, 맨 마지막에 생성하고자 하는 프로젝트명을 써주면 된다.

The screenshot shows the OpenStack Dashboard interface. On the left, there's a sidebar with '프로젝트', '관리', and '인증' sections. The main area is titled '프로젝트' and shows a list of projects. The first project, 'services', is listed under '작업' (Actions). Below it, three more projects are listed: 'project2', 'project1', and 'admin'. The 'project1' row is highlighted with a red dashed border. A large red arrow points from this highlighted row to a terminal window on the right. The terminal window displays the command 'openstack project list' and its output, which matches the data shown in the dashboard. A red box also highlights the 'CLI 환경에서도 확인가능하다!' (Can be confirmed in the CLI environment) message.

ID	Name	Domain Name	Enabled	Action
10cc1ee54ccd4f1d92e11561f670fb07	services	Default	예	작업
3743b4cad934479681a21bdc45aad8a7	project2	Default	예	멤버 관리
6610b67da3cd4e5f8116ea7537d9c389	project1	Default	예	멤버 관리
d103d28082ff4ebca40a7c35912db76b	admin	Default	예	멤버 관리

```
[root@controller ~]# openstack project list
+-----+-----+
| ID      | Name   |
+-----+-----+
| 10cc1ee54ccd4f1d92e11561f670fb07 | services |
| 3743b4cad934479681a21bdc45aad8a7 | project2 |
| 6610b67da3cd4e5f8116ea7537d9c389 | project1 |
| d103d28082ff4ebca40a7c35912db76b | admin    |
+-----+-----+
```

사용자 계정 생성(Keystone)

2) 사용자 계정 생성 명령어

- 사용자 생성 : openstack user1 create --domain default --password-prompt user1

➤ 이 뒤에 user1에 넣고 싶은 password를 입력해주면 완성된다.

The screenshot shows the OpenStack Keystone dashboard under the 'User' tab. A red box highlights the 'CLI 환경에서도 확인가능하다!' (Confirmed in the CLI environment!) message. A red arrow points from the 'user1' entry in the user list to the CLI output. The CLI output shows the details of the newly created user 'user1'.

Field	Value
description	cirros_user
domain_id	default
email	
enabled	True
id	989ba5bd88474aaf8191d7ec159afbfb
name	user1
options	{}
password_expires_at	None

주의! 역할 리스트로 들어가서 _member_로 지정해주자!

역할 할당 명령어 : openstack role add --project project1 --user user1 _member_

Neutron 서비스 구현

Neutron 이란?

- Neutron은 네트워크과 IP 주소를 관리하기 위해 사용되는 오픈스택 네트워크 시스템이다. 오픈스택 네트워크는 클라우드 시스템 설치 시에 클라우드 시스템 설치 시 네트워크가 장애물이나 제약 요소 아니라는 사실을 보장해주고 사용자들이 네트워크 설정을 스스로 할 수 있게 도와주는 역할을 한다.
- Floating IP는 트래픽을 동적으로 IT기반 시설 내부의 다른 자원으로 라우팅시키는 역할이며, 여기서는 대규모 네트워크 지원을 위해 SDN을 사용할 수도 있다. 이 외에도 침입탐지시스템(IDS), 로드밸런싱(LB), 방화벽, VPN(가상 사설망) 등등의 프레임워크를 추가제공 한다.

시스템 정보

서비스 Compute 서비스 블록 스토리지 서비스 네트워크 에이전트

필터

6 항목 표시

유형	이름	호스트	Zone	Status	State	마지막 업데이트됨	작업
Metering agent	neutron-metering-agent	controller	-	활성화됨	Up	0분	
L3 agent	neutron-l3-agent	controller	nova	활성화됨	Up	0분	라우터 보기
Open vSwitch agent	neutron-openvswitch-agent	controller	-	활성화됨	Up	0분	
Metadata agent	neutron-metadata-agent						
DHCP agent	neutron-dhcp-agent						

[root@controller ~ (keystone_admin)]# openstack network agent list

ID	Agent Type	Host	Availability Zone	Alive	State	Binary
11476eb8-8fdf-4e28-b684-e31601f1ffaf	Metering agent	controller	None	: -)	UP	neutron-metering-agent
56cf7090-e255-4d3a-b4db-0f753e0f101e	L3 agent	controller	nova	: -)	UP	neutron-l3-agent
7182e4d4-75ec-45d7-b943-d8b1696db01c	Open vSwitch agent	controller	None	: -)	UP	neutron-openvswitch-agent
88527af0-b8b9-444d-a89a-65c763178a61	Metadata agent	controller	None	: -)	UP	neutron-metadata-agent
af00cc92-6531-4de0-a00c-100844645a34	DHCP agent	controller	nova	: -)	UP	neutron-dhcp-agent
fa86b5b5-7ce4-43d3-9ade-ed1a8a35a2ad	Open vSwitch agent	compute	None	XXX	UP	neutron-openvswitch-agent

6 항목 표시

Neutron 서비스 상태

네트워크 설정(Neutron)

1) 외부 네트워크 설정

- 외부 네트워크 생성 명령어 :

```
openstack network create --project admin --external --provider-network-type flat --provider-physical-network extnet external1
```

➤ 외부 네트워크는 언제나 admin에서 만들어준다. flat 방식의 외부 네트워크를 만들 것.

프로젝트 > 네트워크 > 네트워크

The screenshot shows the 'Neutron' section of the OpenStack dashboard. At the top, there are search and filter options, along with buttons for creating a new network ('+ 네트워크 생성') and deleting an existing one ('네트워크 삭제'). Below this is a table listing two networks:

Name	관련 서브넷	공유	외부	Status	관리 상태	가용성 존	작업
external1	external1_subnet 10.1.20.0/24	아니오	예	Active	UP	nova	네트워크 편집
internal0	internal_subnet0 192.168.2.0/24	아니오	아니오	Active	UP	nova	네트워크 편집

A red dashed box highlights the 'external1' row, indicating it is the selected or focused item.

- 외부 네트워크 서브넷 생성 명령어 :

```
openstack subnet create --network external1 --no-dhcp --allocation-pool start=10.1.20.50,end=10.1.20.100  
--subnet-range 10.1.20.0/24 --gateway=10.1.20.1 external1_subnet
```

➤ Pool은 앞서 설계했던 것처럼 50~100까지로 정해준다.

네트워크 설정(Neutron)

2-1) 내부 네트워크 설정

- 내부 네트워크 생성 명령어 :

```
openstack network create -internal internal1
```

➤ 내부 네트워크는 언제나 해당 사용자 계정에서 만들어준다. Internal1의 경우 project1의 user1이 된다.

관리 > 네트워크 > 네트워크

네트워크									
		Project =		필터		+ 네트워크 생성		삭제	
프로젝트	네트워크 이름	관련 서브넷	DHCP 에이전트	공유	외부	Status	관리 상태	가용성 존	작업
project1	internal1	internal1_subnet 192.168.0.0/24	1	아니오	아니오	Active	UP	nova	네트워크 편집
admin	external1	external1_subnet 10.1.20.0/24	1	아니오	예	Active	UP	nova	네트워크 편집
admin	internal0	internal_subnet0 192.168.2.0/24	1	아니오	아니오	Active	UP	nova	네트워크 편집
project2	internal2	internal2_subnet 192.168.1.0/24	1	아니오	아니오	Active	UP	nova	네트워크 편집

- 내부 네트워크 서브넷 생성 명령어 :

```
openstack subnet create --dhcp --subnet-range 192.168.0.0/24 --network internal1 internal1_subnet
```

➤ 내부 네트워크는 따로 범위가 필요하지 않음. 내부에서 쓰는 IP대역을 전부 이용한다.

네트워크 설정(Neutron)

2-2) 플로팅 IP 만들어주기

- 프로젝트 > 네트워크 > Floating IP > 프로젝트에 IP 할당

➤ 각 사용자 계정마다 만들어준다. 앞으로 만들어질 인스턴스들에게 각각 할당해 줄 예정이기에!

The image shows two screenshots of the OpenStack Neutron interface. The top screenshot is a modal window titled "Floating IP 할당" (Assign Floating IP). It has a dropdown menu "Pool" set to "external1". A text input field "설명" (Description) contains "CentOS7_IP". Below this is a section titled "프로젝트 Quotas" (Project Quotas) showing a progress bar for "유동 IP" (Floating IP) usage at 0 out of 50. A large blue curved arrow points from this window down to the bottom screenshot. The bottom screenshot is a list titled "Floating IP" showing one item: "1 항목 표시" (1 item displayed). The table columns are "IP 주소" (IP Address), "설명" (Description), "Fixed IP 주소 맵핑함" (Fixed IP Address Mappings), "Pool", "Status", and "작업" (Action). The single item listed is "192.168.73.109" with "CentOS7_IP" in the description, an empty mapping list, "external1" in the pool, and "Down" status. The "작업" column for this row has a dropdown menu with "연결" (Associate) selected.

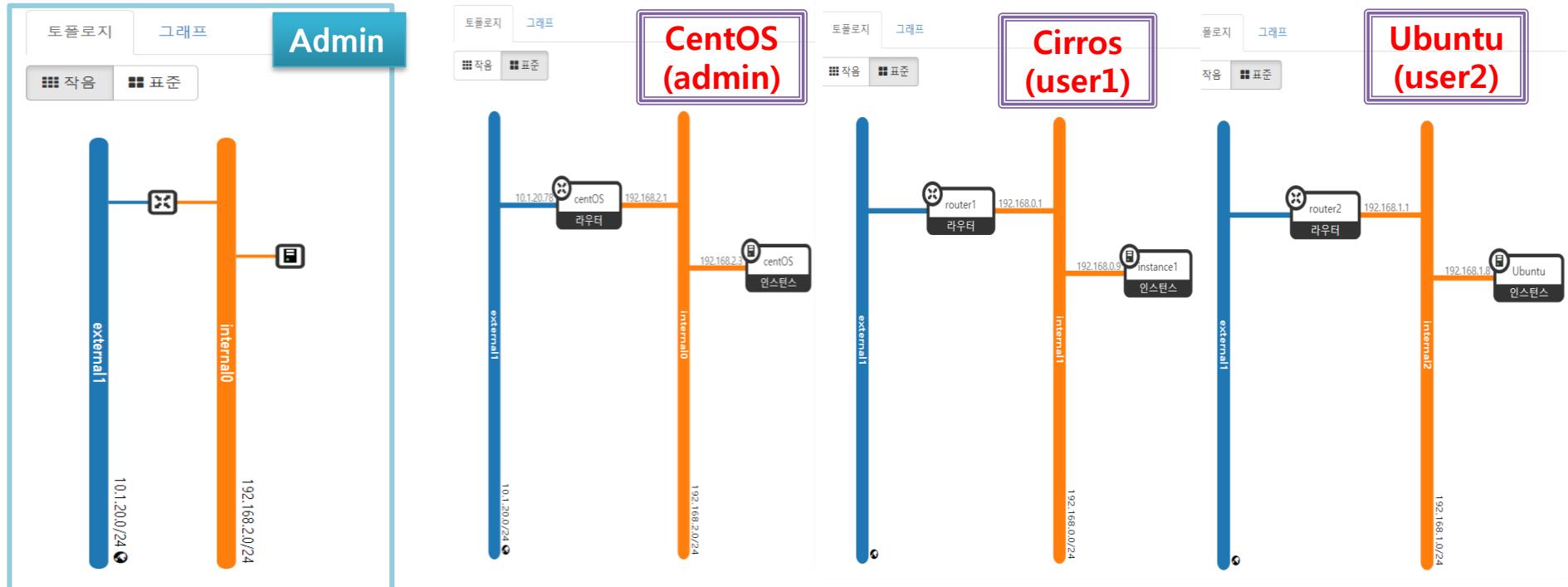
IP 주소	설명	Fixed IP 주소 맵핑함	Pool	Status	작업
192.168.73.109	CentOS7_IP	-	external1	Down	연결

네트워크 설정(Neutron)

3) 전체 프로젝트 네트워크 토플로지

반드시 확인할 것!

- 라우터가 없거나 게이트웨이 연결이 안 되어있을 시
엔 토플로지가 끊어져 있는 모습을 볼 수 있음.

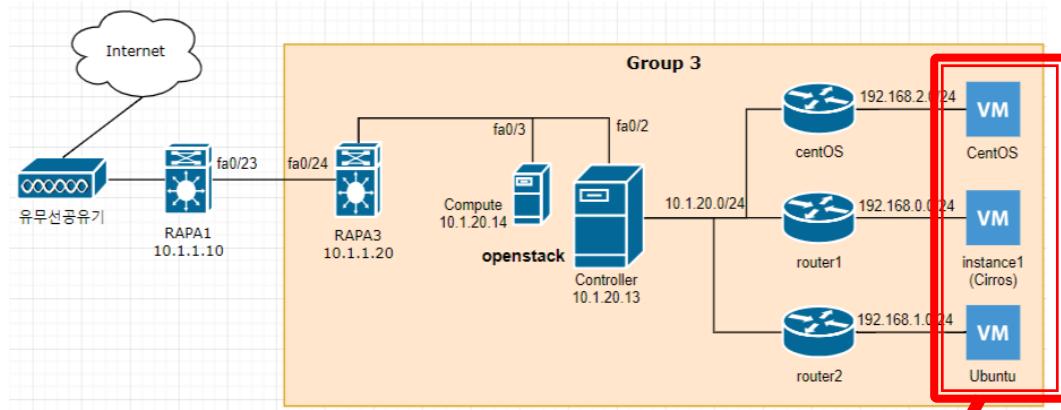


```
[root@controller ~ (keystone_admin)]# openstack network list
```

ID	Name	Subnets
5902a8f1-5d91-42cf-bb90-334e898ae462	internal1	e8db8477-cf82-435a-8634-42eea9c304e8
860dba4e-2345-4a7e-a550-1725c0cc1fa3	external1	5e039ddd-de88-4468-b50d-0beb5fe34f5f
db096252-80bf-4987-a6ba-27b90b17628a	internal0	c2d9900a-c278-43a2-bd2a-2f402f5d6ec7
ea123613-9046-4c6f-bcc6-158ab6c3648c	internal2	09786434-aa3f-4b80-a142-088c12d47fa6

오픈스택 서비스 사용하기

- Glance / Cinder / Nova



프로젝트명	사용자명	인스턴스명	볼륨	이미지	VM IP
admin	admin	CentOS	data-volume	centOS	192.168.2.3
project1	user1	instance1		Cirros	192.168.0.9
project2	user2	Ubuntu		ubuntu	192.168.1.8

Glance

1. 이미지 생성

- CentOS : CentOS-7-x86_64-Minimal-1804.iso
- Cirros : cirros-0.3.4-x86_64-disk.img
- Ubuntu : ubuntu-16.04.4-server-amd64.iso

Cinder

2. 볼륨 생성

- 각 이미지에 맞춰 사용할 볼륨 생성
- 인스턴스 만들 때에 볼륨 선택하여 부착

Nova

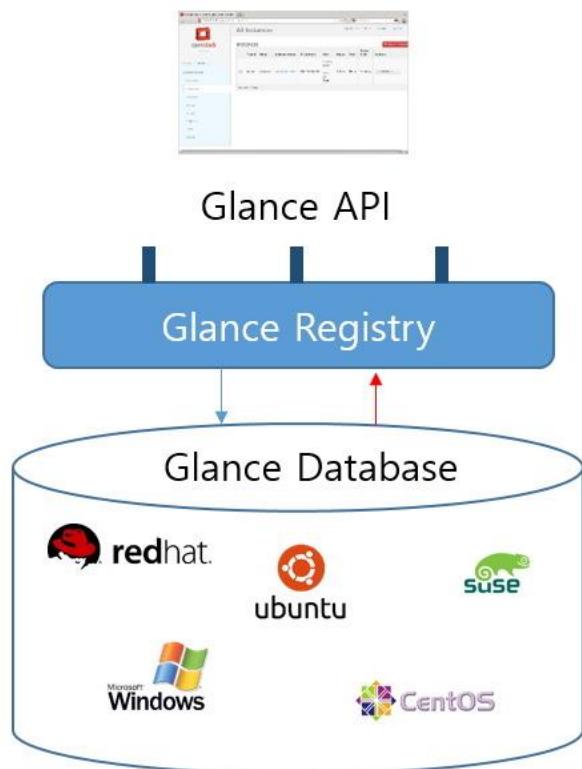
3. 인스턴스(instance) 생성

- 인스턴스 : CentOS, instance1, Ubuntu
- 인스턴스는 네트워크에 따라 각각 IP를 받음.

Glance 서비스 구현

Glance 란?

- 오픈스택 이미지 서비스로 디스크와 인스턴스를 생성할 서버 이미지를 발견해서 등록하고 전송하는 역할을 수행한다. 여기서 저장된 이미지는 마치 템플릿처럼 사용가능하며 무제한으로 백업이 가능하다. Nova 서비스가 이미지에 대한 정보를 제공하고 인스턴스를 생성하기 위해 이미지를 다양하게 설정할 수 있다면, glance 서비스에서는 이 이미지를 추가, 삭제, 공유, 복사할 수 있다.
- 이미지 저장을 위한 DB를 세팅할 수 있다. (MySQL을 사용함)



시스템 정보

서비스	Compute 서비스	블록 스토리지 서비스	네트워크 에이전트
9 항목 표시			
이름	서비스	Region	Endpoints
nova	compute	RegionOne	Admin http://10.1.20.13:8774/v2.1/d103d28082ff4ebca40a7c35912db76b Internal http://10.1.20.13:8774/v2.1/d103d28082ff4ebca40a7c35912db76b Public http://10.1.20.13:8774/v2.1/d103d28082ff4ebca40a7c35912db76b
glance	image	RegionOne	Admin http://10.1.20.13:9292 Internal http://10.1.20.13:9292 Public http://10.1.20.13:9292

Glance 서비스의 구성을 보면 옆의 이미지와 같이 된다.

이미지 생성(Glance)

1) 이미지 생성 명령어

- 이미지 생성 :

```
openstack image create --file=/root/CentOS-7-x86_64-GenericCloud.qcow2
```

```
--disk-format qcow2 --public --unprotected image2
```

> 이미지는 iso 파일 그대로 쓰기보다는 kvm에서 qcow2파일로 만들어 openstack에서 img파일로 다시 만들어서 사용한다. (centOS, ubuntu)

The screenshot shows the OpenStack Dashboard interface. On the left, there's a sidebar with navigation links like '프로젝트', 'API 액세스', 'Compute', '개요', '인스턴스', '이미지' (which is highlighted in blue), '키 페어', '서버 그룹', '블롭', '네트워크', '오브젝트 스토리지', '관리', and '인증'. The main content area is titled '이미지' and lists three images: 'centOS' (ID: 09eadd91-d6d0-4c60-916f-75f0f56078aa), 'cirros' (ID: d437f2d7-4426-4d0d-97d1-ce948a6341d9), and 'ubuntu' (ID: 9f6c8f90-6e32-418c-a168-e353f538688f). Each row includes columns for '이름', '유형', '상태', '가시성', '보호됨', '디스크 포맷', and '크기'. Below the table, there's a '3 항목 표시' link. At the bottom right of the dashboard, there's a terminal window showing the command: [root@controller ~]# openstack image list. The terminal output matches the table above.

ID	Name	Status
09eadd91-d6d0-4c60-916f-75f0f56078aa	centOS	active
d437f2d7-4426-4d0d-97d1-ce948a6341d9	cirros	active
9f6c8f90-6e32-418c-a168-e353f538688f	ubuntu	active

이미지 확인(Glance)

2) 이미지 확인

- 이미지 확인 명령어 :

glance image-show [해당 image의 id]

➤ 이미지는 iso 파일 그대로 쓰기보다는 kvm에서 qcow2파일로 만들어 openstack에서 img파일로 다시 만들어서 사용한다. (centOS, ubuntu)

```
[root@controller ~]# glance image-show d437f2d7-4426-4d0d-97d1-ce948a6341d9
+-----+
| Property      | Value
+-----+
| checksum      | ee1eca47dc88f4879d8a229cc70a07c6
| container_format | bare
| created_at    | 2019-01-03T02:55:01Z
| disk_format   | qcow2
| id            | d437f2d7-4426-4d0d-97d1-ce948a6341d9
| min_disk      | 0
| min_ram       | 0
| name          | cirros
| os_hash_algo  | sha512
| os_hash_value | 1b03ca1bc3fafef448b90583c12f367949f8b0e665685979d95b004e48574b953316799e23240f4f7
|                 | 39d1b5eb4c4ca24d38fdc6f4f9d8247a2bc64db25d6bbdb2
| os_hidden     | False
| owner         | d103d28082ff4ebca40a7c35912db76b
| protected     | False
| size          | 13287936
| status         | active
| tags          | []
| updated_at    | 2019-01-07T05:45:55Z
| virtual_size  | Not available
| visibility    | public
+-----+
```

GUI가 느릴 경우, CLI로 진행하기.

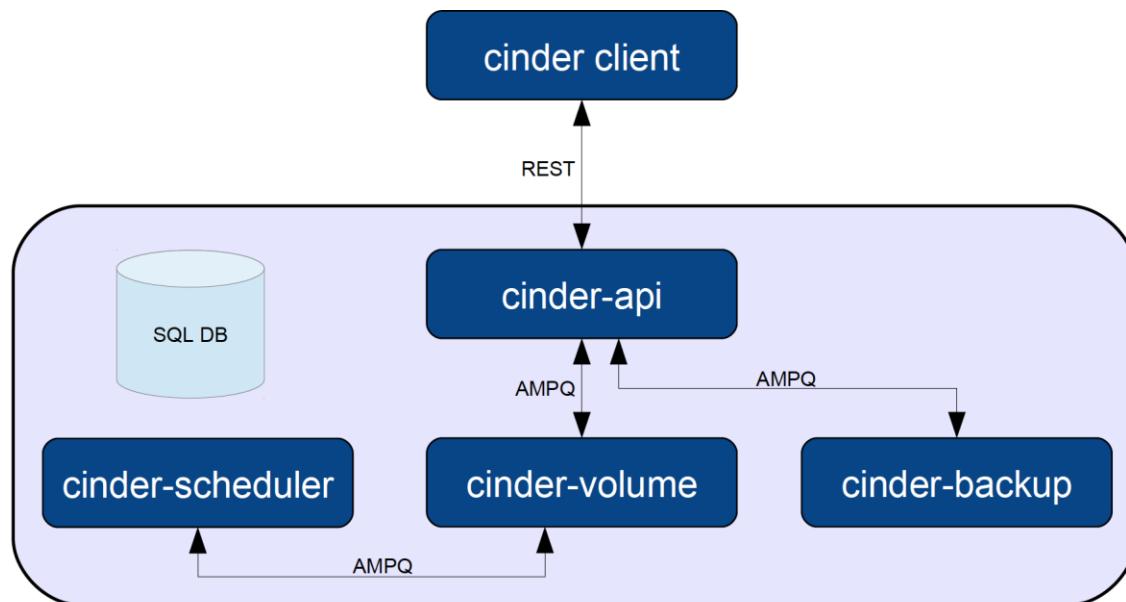
Cinder 서비스 구현

Cinder 란?

- 오픈스택 블록 스토리지 서비스로 Nova 서비스가 제공하는 인스턴스(가상머신)에 지속적으로 사용이 가능한 블록 스토리지 장치를 제공한다. 블록이란 데이터를 저장하기 위한 디스크 블록으로서 파일 시스템에 따라 분리되는 디스크의 영역을 의미하는데, 이런 블록 장치를 생성하고 서버에 부착하고 분리하는 업무를 담당한다. (주로 SAN에서 사용되는 iSCSI를 사용하였다)

관리> 시스템 > 시스템 정보

cinder	volume	RegionOne	Admin http://10.1.20.13:8776/v1/d103d28082ff4ebca40a7c35912db76b
			Internal http://10.1.20.13:8776/v1/d103d28082ff4ebca40a7c35912db76b
			Public http://10.1.20.13:8776/v1/d103d28082ff4ebca40a7c35912db76b



Cinder 서비스 구현

Cinder-api : 여러 클라이언트 서비스로부터 API 요청을 받아 그 요청이 수행되도록 cinder-volume 서비스에 전달한다.

Cinder-scheduler : nova-scheduler 서비스처럼 볼륨을 생성할 수 있는 최선의 스토리지 노드를 선택한다.

Cinder-backup : 블록 스토리지 볼륨을 외부의 스토리지 저장소에 백업한다.

Cinder-volume : cinder-api로부터 받은 요청을 처리한다. VM에게 스토리지 제공을 위해 백엔드에 존재하는 볼륨 디바이스에 대한 읽기, 쓰기 등의 요청에 대해 응답한다.

시스템 정보

서비스 Compute 서비스 블록 스토리지 서비스 네트워크 에이전트 필터

3 항목 표시

이름	호스트	Zone	Status	State	마지막 업데이트됨
cinder-scheduler	controller	nova	활성화됨	Up	0분
cinder-backup	controller	nova	활성화됨	Up	0분
cinder-volume	controller@lvm	nova	활성화됨	Up	0분

3 항목 표시

```
[root@controller ~ (keystone_admin)]# cinder service-list
+-----+-----+-----+-----+-----+
| Binary | Host | Zone | Status | State | Updated_at | Disabled Reason |
+-----+-----+-----+-----+-----+
| cinder-backup | controller | nova | enabled | up | 2019-01-07T06:49:04.000000 | - |
| cinder-scheduler | controller | nova | enabled | up | 2019-01-07T06:49:00.000000 | - |
| cinder-volume | controller@lvm | nova | enabled | up | 2019-01-07T06:49:04.000000 | - |
+-----+-----+-----+-----+-----+
```

```
[root@controller ~ (keystone_admin)]# cinder list
+-----+-----+-----+-----+-----+
| ID | Status | Name | Size | Volume Type | Bootable | Attached to |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```

```
[root@controller ~ (keystone_admin)]# cinder service-list
+-----+-----+-----+-----+-----+
| Binary | Host | Zone | Status | State | Updated_at | Disabled Reason |
+-----+-----+-----+-----+-----+
| cinder-backup | controller | nova | enabled | up | 2019-01-07T06:49:04.000000 | - |
| cinder-scheduler | controller | nova | enabled | up | 2019-01-07T06:49:00.000000 | - |
| cinder-volume | controller@lvm | nova | enabled | up | 2019-01-07T06:49:04.000000 | - |
+-----+-----+-----+-----+-----+
```

```
[root@controller ~ (keystone_admin)]# cinder list
+-----+-----+-----+-----+-----+
| ID | Status | Name | Size | Volume Type | Bootable | Attached to |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```

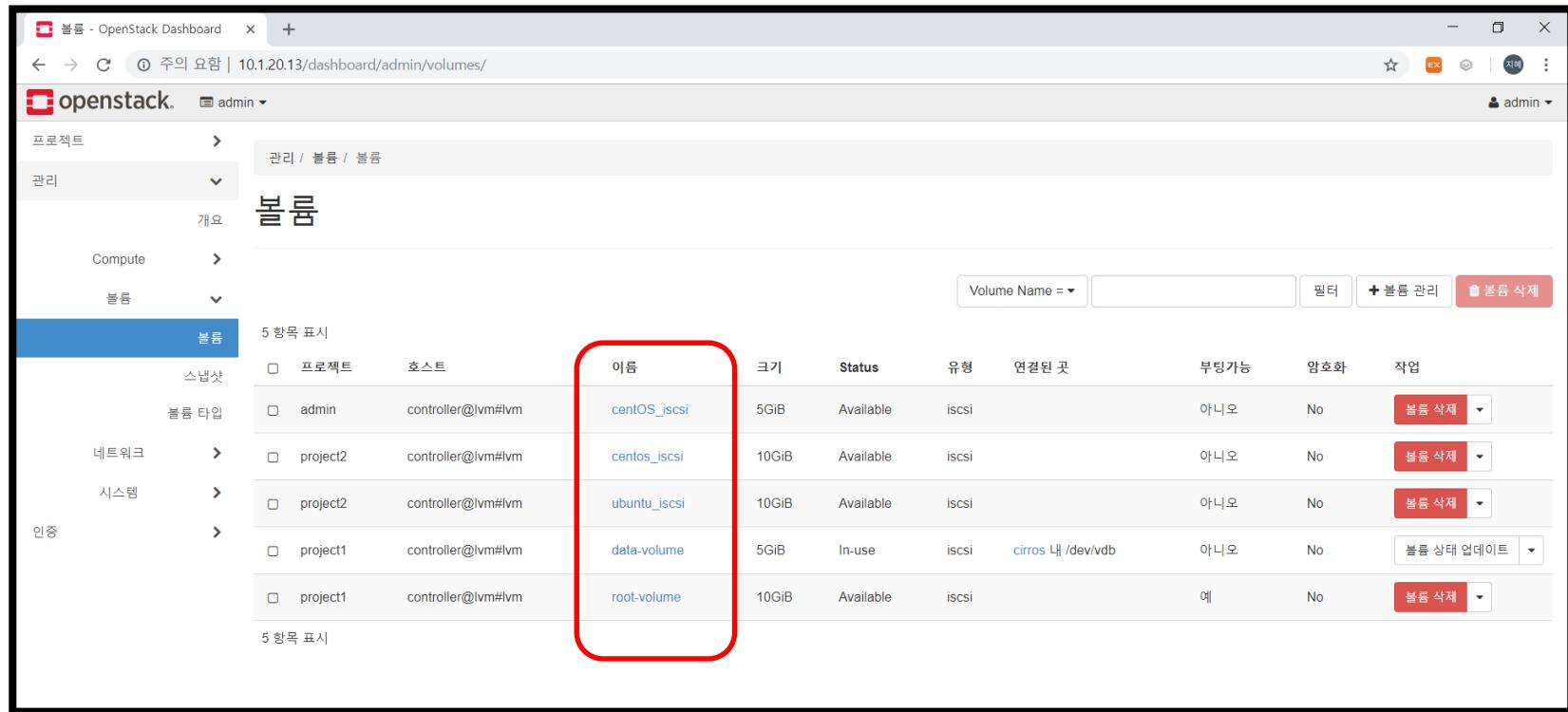
볼륨 생성(Cinder)

1) 볼륨 생성

- 볼륨 생성 명령어 :

openstack volume create --size 10 --type iscsi - image cirros root-volume

- Cirros image에서 iscsi 형식의 사이즈 10의 볼륨을 생성한다.
- 각 이미지마다 하나씩 다 만들어 주기 (iscsi 형식으로)



The screenshot shows the OpenStack Dashboard interface for managing volumes. The left sidebar navigation includes '프로젝트', '관리', '개요', 'Compute', '볼륨' (selected), and '스냅샷'. The main content area is titled '볼륨' and displays a table of volumes. The table columns are: 이름 (Name), 크기 (Size), Status, 유형 (Type), 연결된 곳 (Attached To), 부팅 가능 (Bootable), 암호화 (Encryption), and 작업 (Actions). A red box highlights the '이름' column header. The table data is as follows:

이름	크기	Status	유형	연결된 곳	부팅 가능	암호화	작업
centOS_iscsi	5GiB	Available	iscsi		아니오	No	<button>볼륨 삭제</button>
centos_iscsi	10GiB	Available	iscsi		아니오	No	<button>볼륨 삭제</button>
ubuntu_iscsi	10GiB	Available	iscsi		아니오	No	<button>볼륨 삭제</button>
data-volume	5GiB	In-use	iscsi	cirros 냄 /dev/vdb	아니오	No	<button>볼륨 상태 업데이트</button>
root-volume	10GiB	Available	iscsi		예	No	<button>볼륨 삭제</button>

볼륨 확인(Cinder)

2) 볼륨 확인 및 연결

- 볼륨 확인 명령어 :

openstack volume list

➤ 볼륨이 생성된 계정으로 들어가서 확인해야 있는지 확인 가능하다. 나중에 인스턴스가 만들어지면 꼭 연결해줘야 함!

```
[root@controller / (keystone_user1)]# openstack volume list
+-----+-----+-----+-----+
| ID      | Name     | Status  | Size   | Attached to
+-----+-----+-----+-----+
| 60b67e6e-85ea-4ddb-bc42-3f5901b68eb1 | data-volume | in-use | 5 | Attached to cirros on /dev/vdb |
+-----+-----+-----+-----+
```

현재 cirros에 연결되어있음을 알 수 있음.

The screenshot shows the OpenStack Horizon dashboard with the 'admin' user logged in. The left sidebar has 'admin' at the top, followed by '프로젝트', '관리', '개요', 'Compute', '블룸', and '인증'. Under '블룸', the '볼륨' tab is selected. The main area shows a table titled '볼륨' with one item:

스냅샷	프로젝트	호스트	이름	크기	Status	유형	연결된 곳	부팅 가능	암호화	작업
	project1	controller@lvm#lvm	data-volume	5GiB	In-use	iSCSI	cirros 내 /dev/vdb	아니오	No	<button>볼륨 상태 업데이트</button>

Nova 서비스 구현

Nova 란?

- 오픈스택 컴퓨터 서비스로 IaaS 시스템의 핵심 서비스로서 클라우드 컴퓨팅 운영체제에서 리눅스의 커널과 같은 역할을 담당하고 있다. 오픈스택의 많은 서비스 중에서 일종의 컨트롤러와 같은 역할을 맡고 있다. 보안그룹 및 규칙 생성부터

The screenshot shows the OpenStack Horizon dashboard. The left sidebar navigation includes '프로젝트' (Project), '관리' (Management), '개요' (Overview), 'Compute' (Compute), '네트워크' (Network), '시스템' (System), '기본' (Basic), '메타데이터 정의' (Metadata Definition), and '인증' (Authentication). The 'Compute' section is expanded, showing '서비스' (Services) selected. Below this, it lists 'Compute 서비스' (Compute Services), '블록 스토리지 서비스' (Block Storage Services), and '네트워크 에이전트' (Network Agents). The main content area displays '시스템 정보' (System Information) with a table showing service details:

이름	호스트	Zone	Status	State	마지막 업데이트됨
nova-conductor	controller	internal	활성화됨	Up	0분
nova-scheduler	controller	internal	활성화됨	Up	0분
nova-consoleauth	controller	internal	활성화됨	Up	0분
nova-compute	controller	nova	활성화됨	Up	0분

At the bottom of the page, there is a terminal window displaying the command output:

```
[root@controller ~ (keystone_admin)]# openstack compute service list
```

ID	Binary	Host	Zone	Status	State	Updated At
4	nova-conductor	controller	internal	enabled	up	2019-01-08T08:00:51.000000
5	nova-scheduler	controller	internal	enabled	up	2019-01-08T08:00:53.000000
7	nova-consoleauth	controller	internal	enabled	up	2019-01-08T08:00:51.000000
8	nova-compute	controller	nova	enabled	up	2019-01-08T08:00:52.000000
9	nova-compute	compute	nova	enabled	up	2019-01-08T08:00:43.000000

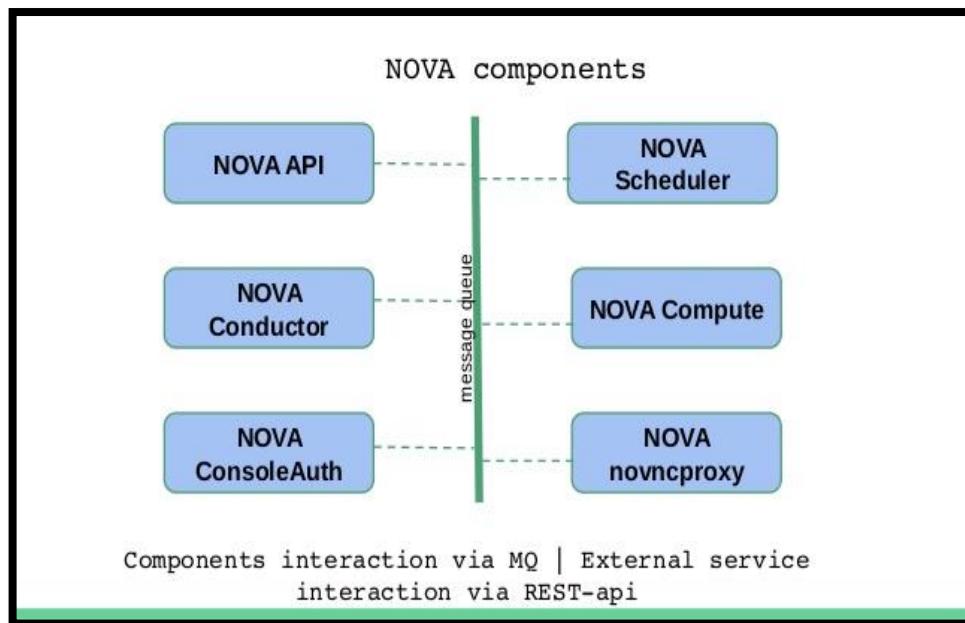
Nova 서비스 구현

nova-conductor : nova-compute와 데이터베이스 사이에서 상호 연동을 중재하는 역할

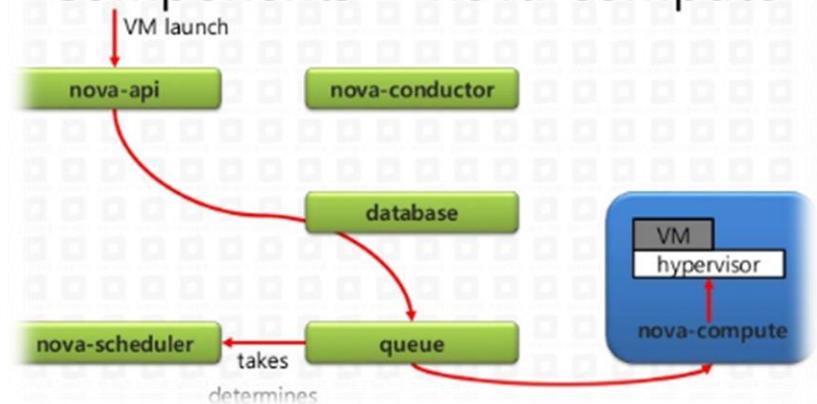
nova-scheduler : 메시지 큐에서 VM 인스턴스 요청을 받으면 어느 컴퓨터 서버 호스트에서 실행할지 결정하는 서비스

nova-consoleauth : 가상머신으로의 콘솔 접속 시 사용자에 대한 인증 토큰을 제공

nova-compute : KVM이나 QEMU 등이 사용하는 libvirt와 같은 하이퍼 바이저 API를 통한 인스턴스(VM) 관리하기 위해 사용되는 데몬



Components > nova-compute



관리보안 생성(Nova)

1) 관리보안 그룹 및 규칙

- 보안그룹 생성 및 보안규칙 생성 명령어:

```
openstack security group create "set0"
```

```
openstack security group rule create --protocol tcp --dst-port 80 --ingress "set0"
```

```
openstack security group rule create --protocol tcp --dst-port 22 --ingress "set0"
```

```
openstack security group rule create --protocol icmp --ingress "set0"
```

➤ Admin계정에서 set0이라는 보안그룹 안에 80번 HTTP포트, 22번 SSH포트, ICMP 전체 포트 열기.

➤ 보안그룹을 하나 만들 때마다 iptable에 전부 기록된다.

각 계정마다 하나씩 다 만들어준다.

관리 보안 그룹 규칙: set0 (7faaa745-7ac9-40b5-be58-780b3c63fc96)

Admin : set0
User1 : set1
User2 : set2

5 항목 표시

Direction	Ether 타입	IP 프로토콜	포트 범위	원격 IP 접두사	원격 보안 그룹
내보냄	IPv4	전체	전체	0.0.0.0/0	-
내보냄	IPv6	전체	전체	::/0	-
들어옴	IPv4	ICMP	전체	0.0.0.0/0	-
들어옴	IPv4	TCP	22 (SSH)	0.0.0.0/0	-
들어옴	IPv4	TCP	80 (HTTP)	0.0.0.0/0	-

5 항목 표시

Flavor 생성(Nova)

2) Flavor 생성

- admin 계정에서 flavor 생성 명령어 :

```
openstack flavor create --id 7 --ram 1024 --disk 15 --vcpus 1 --public m2.small
```

- 각각의 스펙을 그대로 넣어줘 m2.small이라는 flavor하나를 생성한다.
- Flavor란 인스턴스 생성 시 기본적으로 사용할 자원을 정의한 이름이다. (CPU, 메모리, 디스크 용량 등등)

인스턴스 생성(Nova)

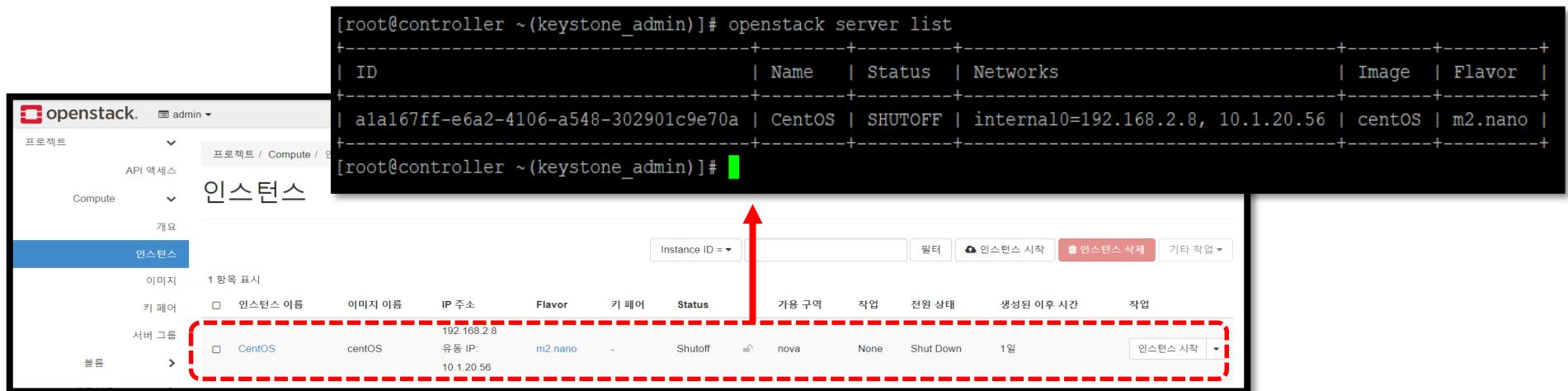
3) 인스턴스 생성

- 각 사용자 계정에서 인스턴스 생성 명령어 :

```
openstack instance1 create --image cirros --flavor m2.nano --nic net-id=[내부 네트워크의 ID] --security-group set1 instance1
```

➤ 위의 예시는 user1 계정에서 만든 instance1로, cirros image 와 m1.nano의 flavor, nic id는 만들어둔 internal1의 id, 보안그룹은 set1으로 만든다.

➤ 이 외에도 여러가지 옵션이 있는데 GUI는 이 옵션들을 하나씩 보여주는 반면, CLI에서는 한번에 만들기에 오타나지 않게 주의!



The screenshot shows the OpenStack Compute dashboard under the 'Instances' tab. A red dashed box highlights the first instance listed, which is a CentOS instance. The instance details shown are:

이미지	1 항목 표시	키 페어	인스턴스 이름	이미지 이름	IP 주소	Flavor	키 페어	Status	가용 구역	작업	전원 상태	생성된 이후 시간	작업
서버 그룹	>	CentOS	centOS		192.168.2.8 유동 IP: 10.1.20.56	m2.nano	-	Shutoff	nova	None	Shut Down	1일	인스턴스 시작

A red arrow points from the CLI output at the top to the highlighted instance in the table.

```
[root@controller ~ (keystone_admin)]# openstack server list
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | Status | Networks | Image | Flavor |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| a1a167ff-e6a2-4106-a548-302901c9e70a | CentOS | SHUTOFF | internal0=192.168.2.8, 10.1.20.56 | centos | m2.nano |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
[root@controller ~ (keystone_admin)]#
```

→ Admin 계정의 CentOS 인스턴스를 CLI 환경에서도 똑같이 확인 가능하다.

각각의 사용자 계정으로 keystone 인증한 뒤에 만들기 가능.

keystonerc_admin - centOS
keystonerc_user1 - instance1(cirros)
keystonerc_user2 - ubuntu

Ex)

```
# source /root/keystonerc_user2
[root@controller ~ (keystone_user2)]#
```

Floating IP 연결(Nova)

4) Floating IP 연결

- 각 사용자 계정에서 플로팅 IP 연결 명령어 :

Openstack **instance1** add floating ip **instance1** [만들어둔 floating IP]

➤ 각 사용자 계정에서 미리 만들어둔 플로팅 IP를 하나씩 할당, 연결해준다.

IP 주소	설명	Fixed IP 주소 맵핑함	Pool	Status	작업
192.168.73.109	CentOS7_IP	-	external1	Down	<button>연결</button>

주의 : 플로팅 IP가 할당되지 않으면 외부에서 인스턴스에 접근이 불가하다!

인스턴스 확인(Nova)

5) 플로팅 IP까지 할당 후, 전체 인스턴스 확인

The screenshot shows the Nova Compute interface under the 'Instances' tab. A single instance named 'CentOS' is listed, showing its details: Image ID (1 황목 표시), Key Pair (None), Server Group (None), and Flavor (m2.nano). The instance has an IP address of 192.168.2.8 and a floating IP of 10.1.20.56. It is currently in a 'Shutoff' state and assigned to the nova host. The status bar indicates it was created 1 day ago.

인스턴스 이름	이미지 이름	IP 주소	Flavor	키 페어	Status	가용 구역	작업	전원 상태	생성된 이후 시간	작업
CentOS	centOS	192.168.2.8 유동 IP: 10.1.20.56	m2.nano	-	Shutoff	nova	None	Shut Down	1일	인스턴스 시작

→ Admin 계정의 CentOS 인스턴스

The screenshot shows the Nova Compute interface under the 'Instances' tab. A single instance named 'cirros' is listed, showing its details: Image ID (cirros), Key Pair (None), Server Group (None), and Flavor (m2.nano). The instance has an IP address of 192.168.0.9 and a floating IP of 10.1.20.53. It is currently in a 'Shutoff' state and assigned to the nova host. The status bar indicates it was created 5 days ago.

인스턴스 이름	이미지 이름	IP 주소	Flavor	키 페어	Status	가용 구역	작업	전원 상태	생성된 이후 시간	작업
cirros	cirros	192.168.0.9 유동 IP: 10.1.20.53	m2.nano	-	Shutoff	nova	None	Shut Down	5일	인스턴스 시작

→ user1 계정의 cirros 인스턴스

The screenshot shows the Nova Compute interface under the 'Instances' tab. A single instance named 'ubuntu' is listed, showing its details: Image ID (ubuntu), Key Pair (None), Server Group (None), and Flavor (m2.nano). The instance has an IP address of 192.168.1.21 and a floating IP of 10.1.20.54. It is currently in an 'Active' state and assigned to the nova host. The status bar indicates it was created 23시간, 26분 ago.

인스턴스 이름	이미지 이름	IP 주소	Flavor	키 페어	Status	가용 구역	작업	전원 상태	생성된 이후 시간	작업
ubuntu	ubuntu	192.168.1.21 유동 IP: 10.1.20.54	m2.nano	-	Active	nova	None	Running	23시간, 26분	스냅샷 생성

→ user2 계정의 ubuntu 인스턴스

하이퍼바이저(Nova)

6) 하이퍼바이저 확인

- admin에서 하이퍼바이저 확인

관리 > Compute > 하이퍼바이저

➤ Controller와 Compute의 각 상태를 알려준다.

하이퍼바이저 - OpenStack Dashboard

관리 / Compute / 모든 하이퍼바이저

모든 하이퍼바이저

하이퍼바이저 요약

- 호스트 집합
- 인스턴스
- Flavor
- 이미지
- 블롭
- 네트워크
- 시스템

하이퍼바이저

Compute 호스트

호스트 이름	유형	VCPUs (사용 중)	VCPUs (전체)	RAM (사용 중)	RAM (전체)	로컬 저장소 (사용 중)	로컬 저장소 (전체)	인스턴스
compute	QEMU	0	4	512MB	7.9GB	0바이트	49GB	0
controller	QEMU	2	4	1.5GB	7.9GB	16GB	49GB	2

2 항목 표시					
호스트	가용성 존	Status	State	업데이트 이후 시간	작업
controller	nova	활성화됨	Up	0분	<button>서비스 사용안함</button>
compute	nova	활성화됨	Up	0분	<button>서비스 사용안함</button>

```
[root@controller ~ (keystone_admin)]# nova hypervisor-list
+-----+-----+-----+-----+
| ID          | Hypervisor hostname | State | Status |
+-----+-----+-----+-----+
| 373e0ddf-c056-4e6d-b847-39dd053343dc | controller           | up    | enabled |
| ffc9ae79-e5b2-4566-bb13-00fd27cd5c12 | compute              | up    | enabled |
+-----+-----+-----+-----+
```

추가사항(Nova)

7) CLI에서 인스턴스 확인 및 하이퍼바이저 상세정보 확인

- 각 인스턴스는 각 keystone 토큰에서 만들어줬기에 각 계정에 들어가서만 확인 가능하다. 그리고 하이퍼바이저 상세정보 또한 알아볼 수 있다.

Nova list / nova hypervisor-stats

```
[root@controller ~ (keystone_admin)]# nova list
+-----+-----+-----+-----+-----+
| ID           | Name    | Status | Task State | Power State | Networks
+-----+-----+-----+-----+-----+
| a1a167ff-e6a2-4106-a548-302901c9e70a | CentOS  | ACTIVE | -          | Running     | internal0=192.168.2.8, 10.1.20.56 |
+-----+-----+-----+-----+-----+
```

→ Admin 계정의 CentOS 인스턴스

```
[root@controller ~ (keystone_admin)]# source /root/keystonerc_user1
[root@controller ~ (keystone_user1)]# nova list
+-----+-----+-----+-----+-----+
| ID           | Name    | Status | Task State | Power State | Networks
+-----+-----+-----+-----+-----+
| 8376e35b-b3e5-4b37-a221-e880751c41ab | cirros  | ACTIVE | -          | Running     | internal1=192.168.0.9, 10.1.20.53 |
+-----+-----+-----+-----+-----+
```

→ user1 계정의 cirros 인스턴스

Hypervisor의 상세정보확인

```
[root@controller ~ (keystone_admin)]# nova hypervisor-stats
+-----+-----+
| Property      | Value |
+-----+-----+
| count         | 2     |
| current_workload | 0     |
| disk_available_least | 61   |
| free_disk_gb | 82   |
| free_ram_mb  | 14204 |
| local_gb     | 98   |
| local_gb_used | 16   |
| memory_mb    | 16252 |
| memory_mb_used | 2048 |
| running_vms  | 2     |
| vcpus         | 8     |
| vcpus_used   | 2     |
+-----+-----+
```

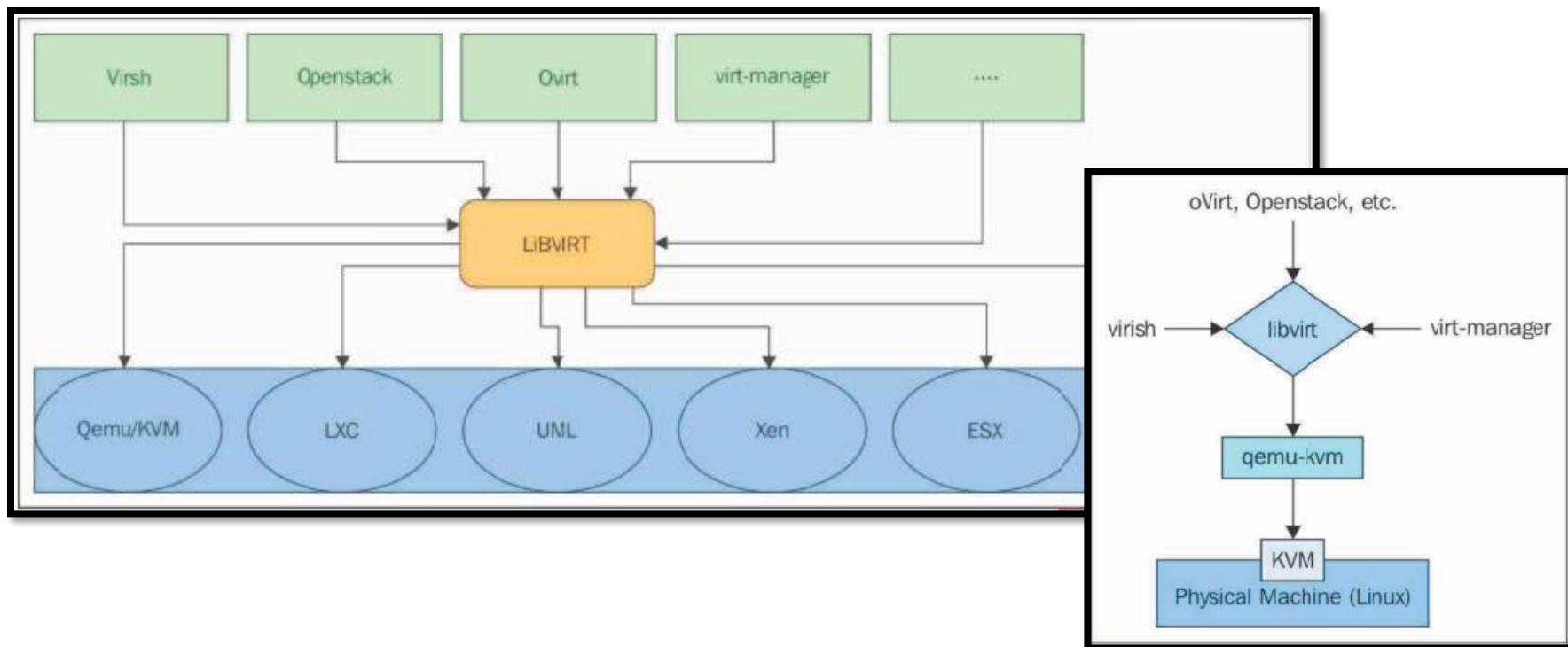
4. 프라이빗 클라우드 서비스 망 구축

- KVM기반

KVM 서비스 구현

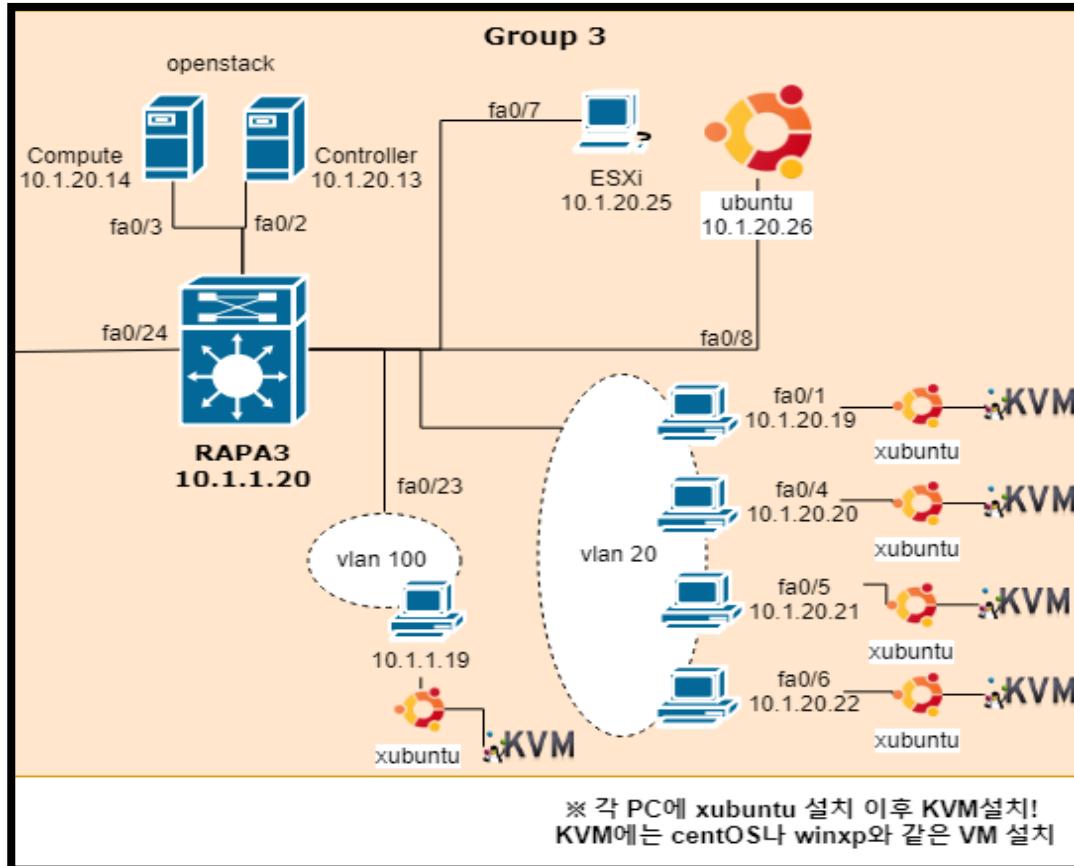
KVM Libvirt란?

- Libvirt는 다양한 하이퍼바이저와 통신이 가능하다. 그 중 KVM/QEMU와도 통신이 가능하며 애플리케이션 프로그래밍 인터페이스(API)로 virsh(커맨드라인 인터페이스 클라이언트)와 virt-manager(그래픽유저 인터페이스 클라이언트)로 구성되어 있다.
- Libvirt는 리눅스의 libvirtd 데몬으로 실행하며 가상머신을 관리하는 관리층이라고 보면 된다. 필자는 Libvirt를 통해 CentOS를 KVM 위에 설치할 것이다.



KVM 전체 구성도

- KVM과 Docker 구성도



1. 구성도 설명

****Ubuntu(xfce)기반으로 KVM 설치진행**

1. Ubuntu(xfce)로 설치 후 진행
 - Ubuntu IP : 10.1.20.26
 - KVM VM(CentOS) : 10.1.20.39
2. KVM으로 구현 후 Docker 설치
3. Docker에서 Cacti Container로 설치
4. Switch SNMP 161번 포트 열어주기
5. NMS로 결과 측정

KVM 하드웨어 구성

- KVM 서버 사양

- CPU : i7 (Dual Core, INTEL-VT 지원)
- Memory : 8Gbyte
- SSD : 1Tbyte
- NIC : 1Gbps
- OS : Ubuntu 16.04 server

KVM 소프트웨어 구성

- KVM 기반 OS : Ubuntu 16.04 server
- 하이퍼바이저 관리도구 : Virtualization Host, KVM_libvirt, virt_manager, virt_viewer 등등…

여기서 잠깐! 꼭 점검해야 되는 조건들!

- 1) SELinux disable로 되어있는지.
- 2) Firewall disable 되어있는지.
- 3) NetworkManager disable 되어있는지.
- 4) External 네트워크 NIC 설정 되어있는지
- 5) DNS 동작을 하는지.
- 6) CPU 가상화 지원을 하는지.

Virsh 의 중요 명령어 리스트

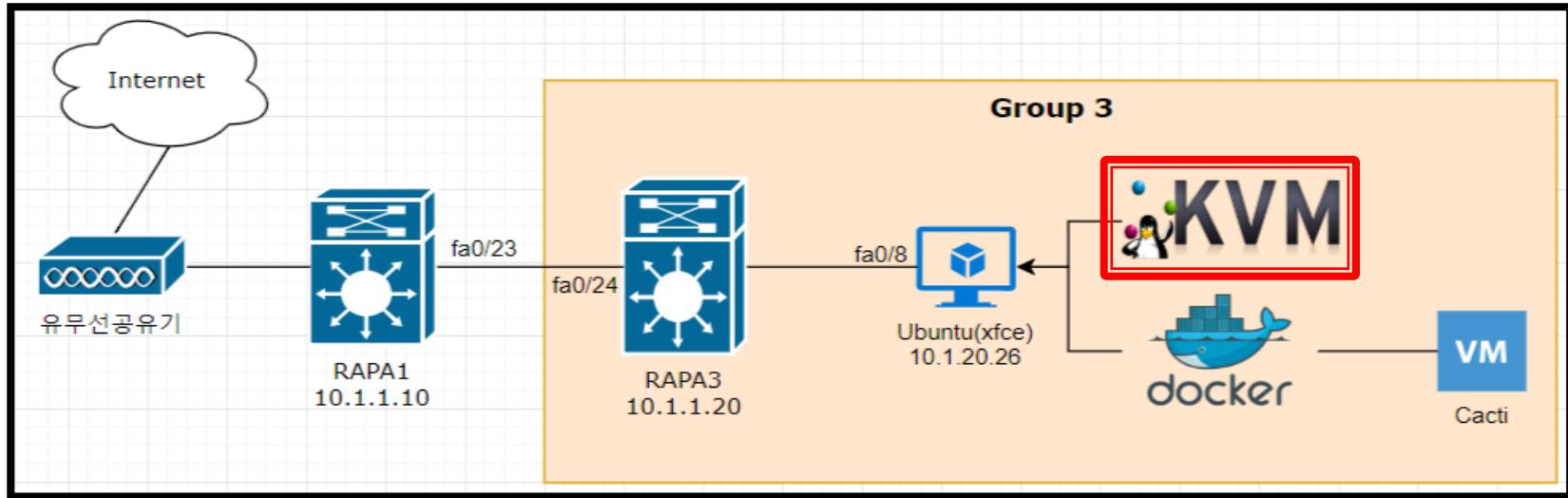
명령	Guest 관리	Guest 모니터링	Host, 하이퍼바이저	가상 네트워크	스토리지
리스트	Start, stop	Memstat, cpustat	Capablilties, nodeinfo	Net-list, net-define	Pool-list, pool-define

KVM 설치

1) KVM 설치

- 먼저 Ubuntu(xfce)를 설치한 후에 그 위로 KVM을 설치한다.

굳이 Ubuntu를 쓰지 않고 CentOS7위에 설치해도 상관없다! (명령어가 조금씩 바뀌는 것에 유의하자)



KVM 설치

1. 가상화 확인

```
egrep -c '(vmx|svm)' /proc/cpuinfo
```

☞ 0이 아닌 숫자가 뜨면 ok!

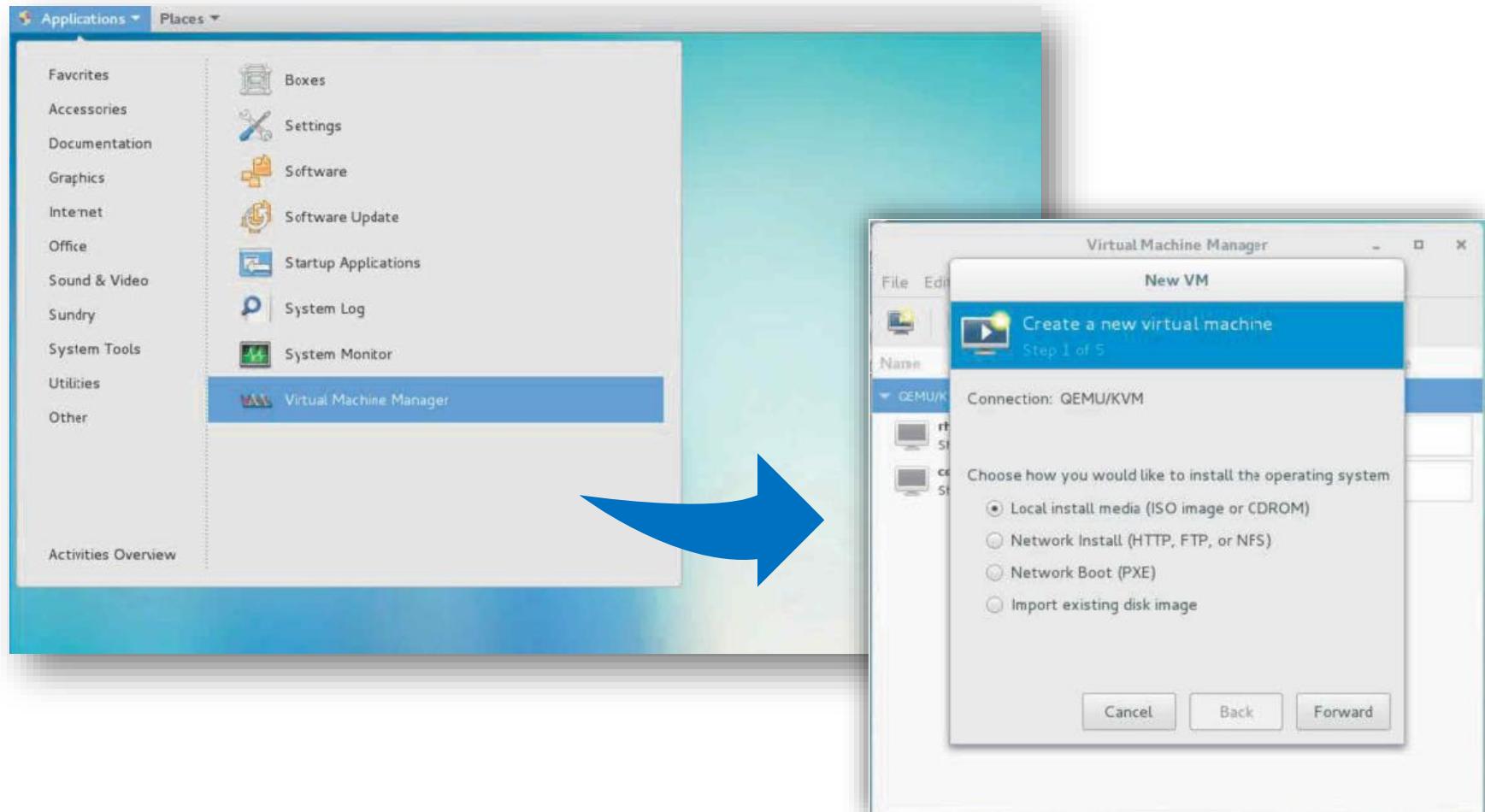
2. KVM 설치

```
sudo apt update  
sudo apt install qemu-kvm libvirt-bin  
ubuntu-vm-builder bridge-utils  
virtinst virt-manager libosinfo-bin
```

KVM 실행

2) KVM Virt-Manager로 VM을리기

-KVM 설치 후 libvirt로 설치한 virtual Machine Manager로 들어가, 새로운 VM을 만든다. (필자는 winXP와 centOS를 설치함)

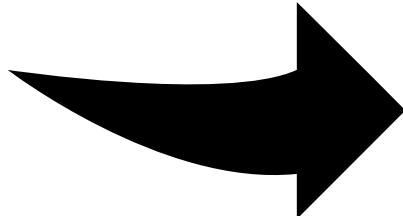


5.NMS/SNMP기반 관제구축

프라이빗 클라우드 서비스 운영(관제)

1) 클라우드 서버 성능 관리

- 오픈 스택 기반 서버 성능 관리
 - CPU 사용률 관리
 - Memory 사용률 관리
 - Swap 사용률 관리
- 오픈스택 서비스 성능 관리



2) 클라우드 네트워크 성능 관리

- 오픈스택 기반 네트워크 성능 관리
 - 트래픽 In/Out 사용률
 - 트래픽 In/Out 사용량
 - 인터페이스 Error 발생률

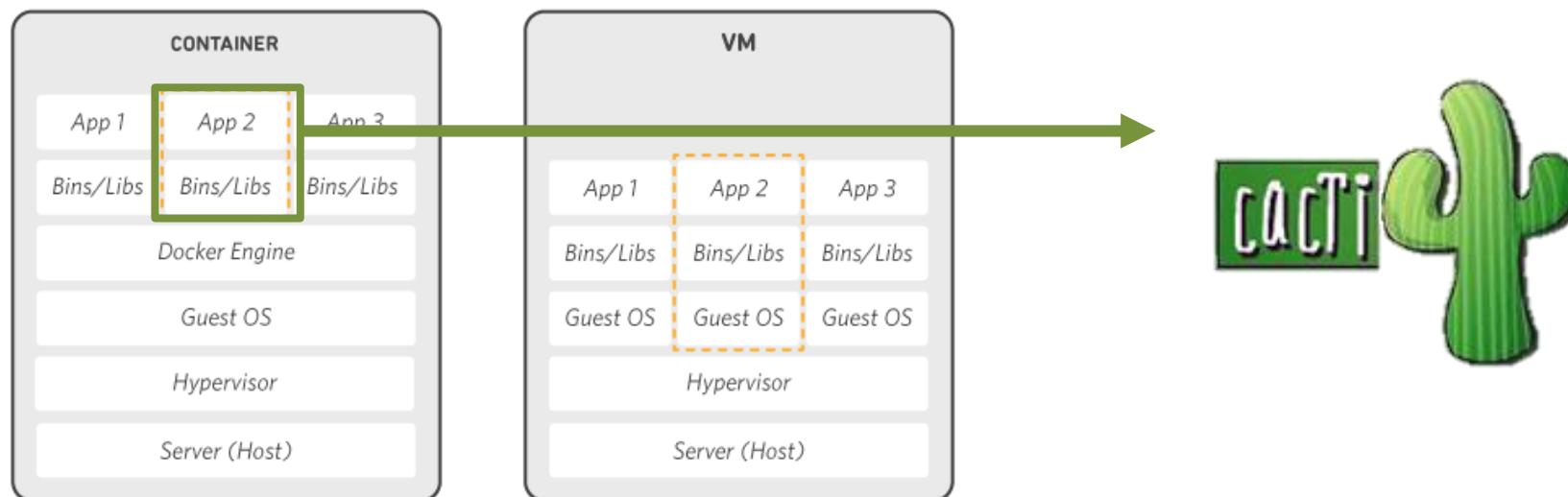
Docker_Cacti 구현

Docker 란?

- 도커는 컨테이너 기반의 오픈소스 가상화 플랫폼으로 다양한 프로그램, 실행환경을 컨테이너로 추상화하고 동일한 인터페이스를 제공하여 프로그램의 배포 및 관리를 단순하게 해준다. 그렇기에 필자는 도커의 컨테이너로 Cacti를 구현하여 한결 가벼운 환경으로 진행할 수 있었다.

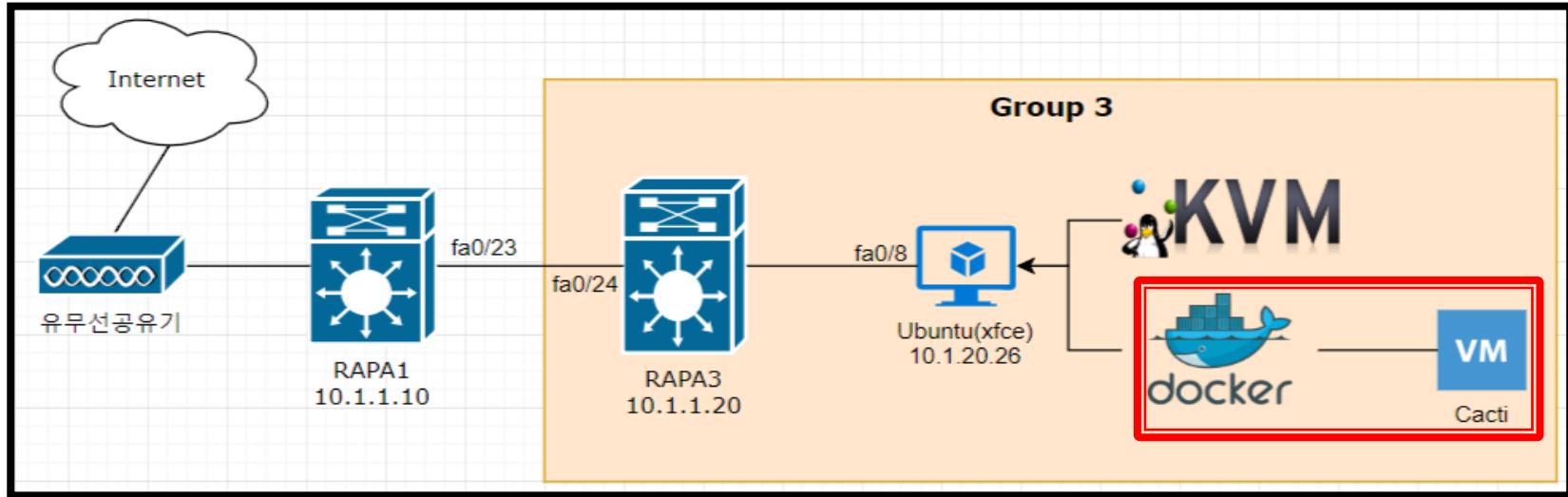
Cacti 란?

- Cacti는 RRDTool, SNMP를 기반으로 한 모니터링 툴로 APM(Apache, PHP, MySQL)과 연계하여 편리하게 사용량을 측정할 수 있다. 이 또한 오픈 소스 프로그램으로 매번마다 프리웨어가 업데이트되고 있으므로 필자는 가장 최신버전으로 Cacti를 구현하였다.



Docker_Cacti 설치

1) Docker / Cacti 설치



3. Docker 설치

```
sudo apt install docker.io
```

4. Cacti 설치 (Container 만들기)

Docker hub에서 cacti 이미지 가져와서 컨테이너 실행

```
sudo docker pull quantumobject/docker-cacti
```

```
sudo docker run -dt -p 8080:80 -p 161:161 --name docker-cacti -e TZ=Asia/Seoul
```

```
quantumobject/docker-cacti
```

Cacti 실행(NMS 측정)

2) 웹에서 Cacti 기본 설정해주기

- http:// [ubuntu_IP]:8080/cacti

- 들어가면 아래와 같은 Cacti 설치 마법사 화면이 나온다.
- 시작하기 전 기본 설정이므로 default 설정으로 next를 누르다 마지막 Template Setup에서 전체 다 설치해준다.

Cacti Installation Wizard

Template Setup

Please select the Device Templates that you wish to use after the Install. If your Operating System is Windows, you need to ensure that you select the 'Windows Device' Template. If your Operating System is Linux/UNIX, make sure you select the 'Local Linux Machine' Device Template.

Device Templates allow you to monitor and graph a vast assortment of data within Cacti. After you select the desired Device Templates, press 'Finish' and the installation will complete. Please be patient on this step, as the importation of the Device Templates can take a few minutes.

Templates			
Name	Description	Author	Homepage
Cisco Router	The Cisco Router Device Package	The Cacti Group	http://www.cacti.net
Generic SNMP Device	The Generic SNMP Device Package	The Cacti Group	http://www.cacti.net
Local Linux Machine	The Local Linux Device Package	The Cacti Group	http://www.cacti.net
Net-SNMP Device	The Net-SNMP Device Package	The Cacti Group	http://www.cacti.net
Windows Device	The Windows Device Template	The Cacti Group	http://www.cacti.net

NOTE: Press 'Finish' to complete the installation process after selecting your Device Templates.

Previous Finish

Cacti 실행(NMS 측정)

3-1) Cacti에 SNMP 올려 측정해보기

- ESXi 를 먼저 테스트용으로 측정하기.

(먼저 ESXi를 SSH 접속하여 SNMP를 열어준다.)

```
esxcli system snmp set -c public  
esxcli system snmp set -e yes  
esxcli system snmp get
```

//community
//enable true로 바꾸는 명령어
//snmp 정보 갖고 오기. 세팅 설정 확인 가능

```
[root@localhost :/etc] esxcli system snmp get  
Authentication:  
Communities: public  
Enable: true  
Engineid: 00000063000000a100000000  
Hwsrc: indications  
Largestorage: true  
Loglevel: info  
Notraps:  
Port: 161  
Privacy:  
Remoteusers:  
Syscontact:  
Syslocation:  
Targets:  
Users:  
V3targets:
```

선택 Telnet 10.1.1.20

```
enrollment selfsigned  
subject-name cn=10S-Self-Signed-Certificate-3799449728  
revocation-check none  
rsakeypair TP-self-signed-3799449728  
!  
RAPA3#sh snmp  
Chassis: CAT1102ZK0R  
4698 SNMP packets input  
0 Bad SNMP version errors  
0 Unknown community name  
0 Illegal operation for community name supplied  
0 Encoding errors  
2566 Number of requested variables  
0 Number of altered variables  
2566 Get-request PDUs  
0 Get-next PDUs  
0 Set-request PDUs  
0 Input queue packet drops (Maximum queue size 1000)  
4698 SNMP packets output  
0 Too big errors (Maximum packet size 1500)  
0 No such name errors  
0 Bad values errors  
0 General errors  
4698 Response PDUs  
0 Trap PDUs  
0 global trap: disabled  
logging: disabled  
t enabled
```

Cacti 실행(NMS 측정)

3-2) Cacti에 SNMP 올려 측정해보기

- Console > Management > Devices > edit

- 아래의 예시는 따로 시험삼아 스위치에 연결한 ESXi를 SNMP로 측정해본 결과이다. 이와 같이 오픈스택과 KVM을 통해 올려본 VM들도 측정해본다.
- 주의사항 : 스위치로 들어가서 RAPA3(config)#snmp-server community public ro 로 161번포트를 열어줘야 한다! (혹은 방화벽 꼬기)

성공했을 시, 위와 같이 SNMP 정보가 뜬다!

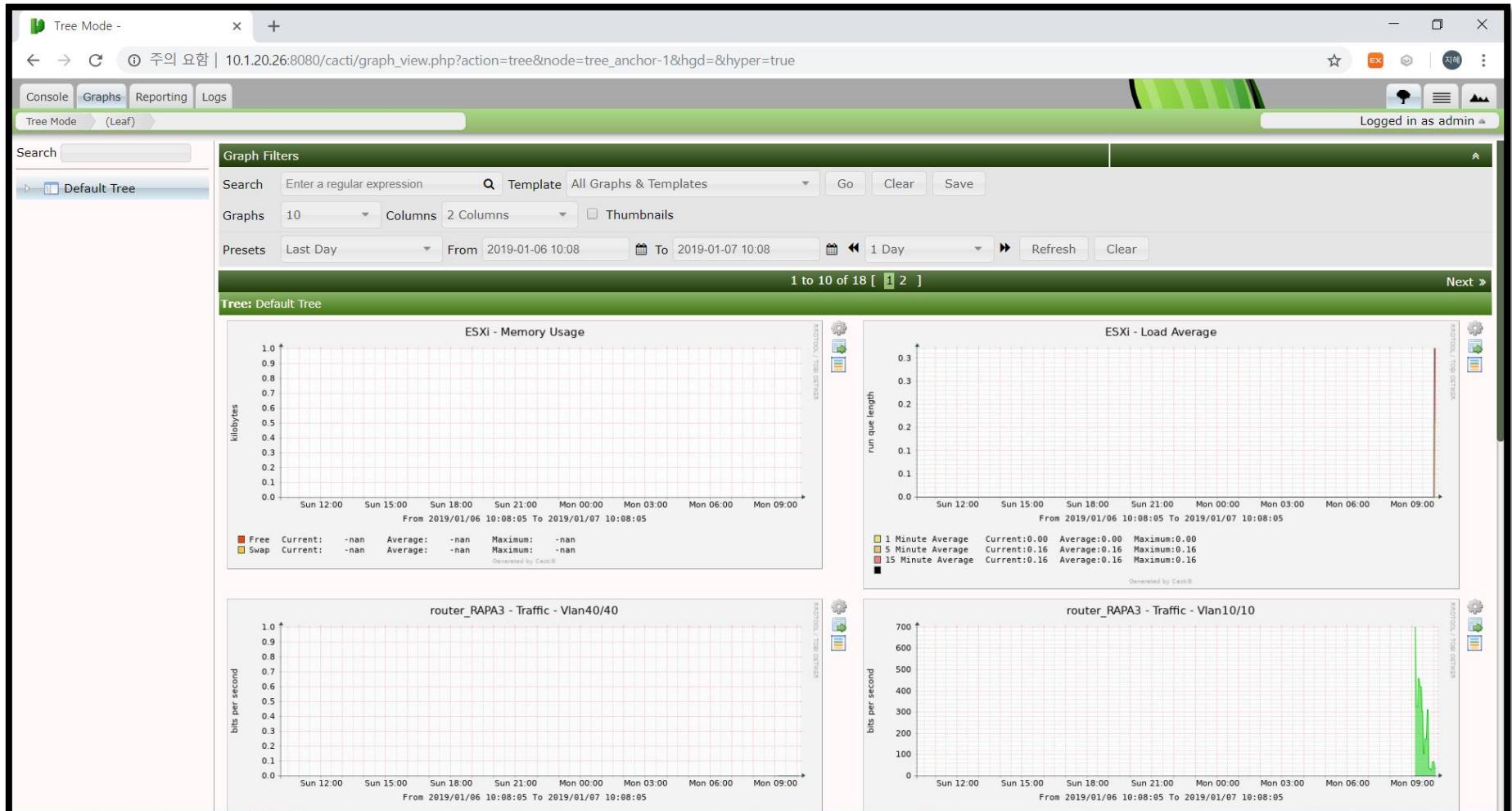
The screenshot shows the Cacti web interface with the following details:

- Left Sidebar:** Contains links for Create, Management, Devices, Sites, Trees, Graphs, Data Sources, Aggregates, Data Collection, Templates, Automation, Presets, Import/Export, Configuration, and Utilities. A cactus icon is displayed.
- Top Bar:** Shows tabs for Untitled Diagram.html, Console -> Devices -> (Edit), and localhost.localdomain - VMware.
- Current Page:** Device [edit: ESXi] (IP: 10.1.20.25). The page includes sections for General Device Options, SNMP Options, Availability/Reachability Options, and a Ping Results section showing a successful ping to 10.1.1.2.
- Right Panel:** Displays a terminal window with the following output:

```
RAPA3#sh snmp
Chassis: CAT1102ZK0R
4698 SNMP packets input
 0 Bad SNMP version errors
 0 Unknown community name
 0 Illegal operation for community name supplied
 0 Encoding errors
2566 Number of requested variables
 0 Number of altered variables
2566 Get-request PDUs
 0 Get-next PDUs
 0 Set-request PDUs
 0 Input queue packet drops (Maximum queue size 1000)
4698 SNMP packets output
 0 Too big errors (Maximum packet size 1500)
 0 No such name errors
 0 Bad values errors
 0 General errors
 4698 Response PDUs
 0 Trap PDUs
SNMP global trap: disabled
SNMP logging: disabled
SNMP agent enabled
```

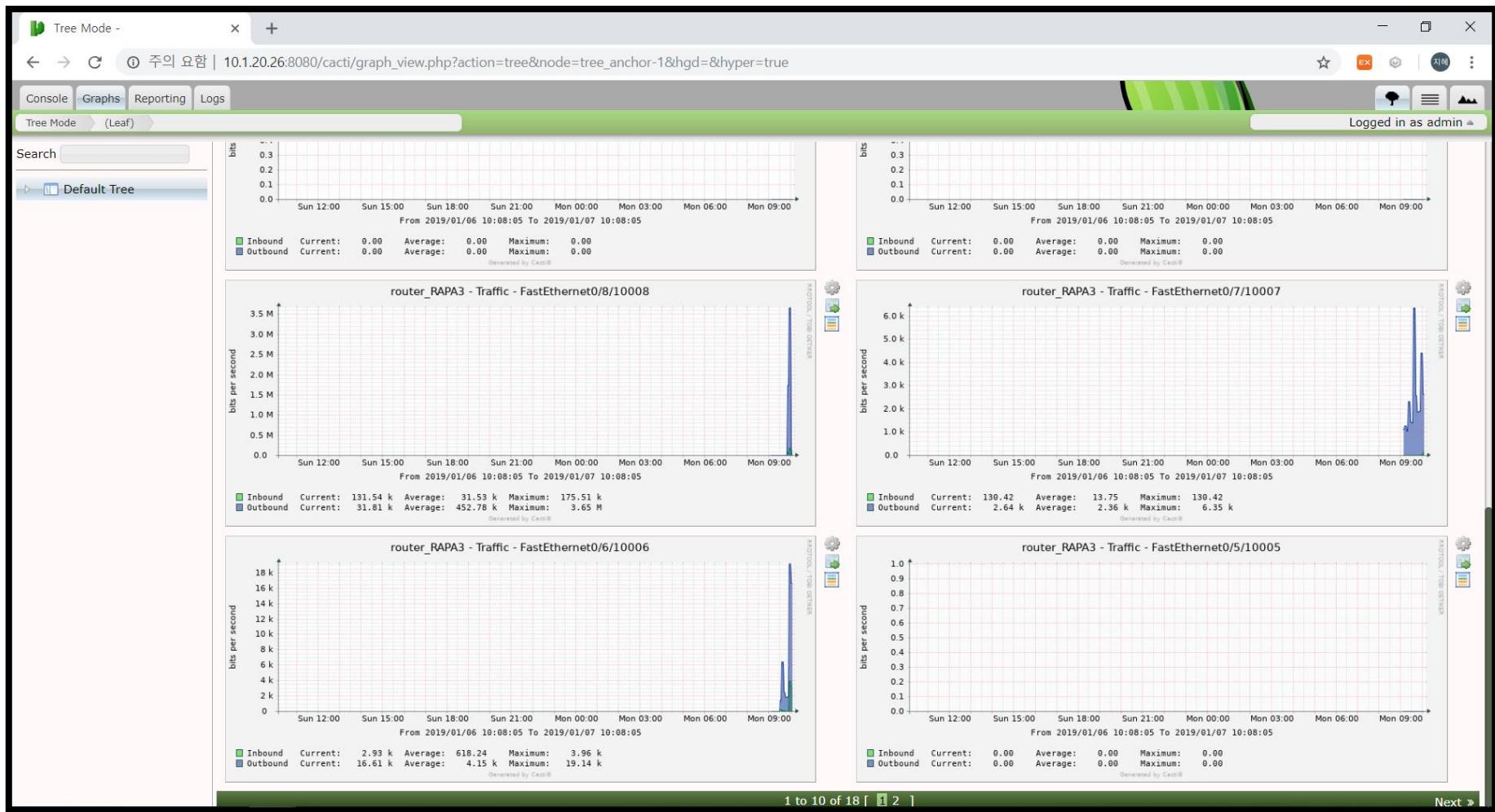
Cacti 실행(NMS 측정)

4) Cacti 측정결과(switch_RAPA3)



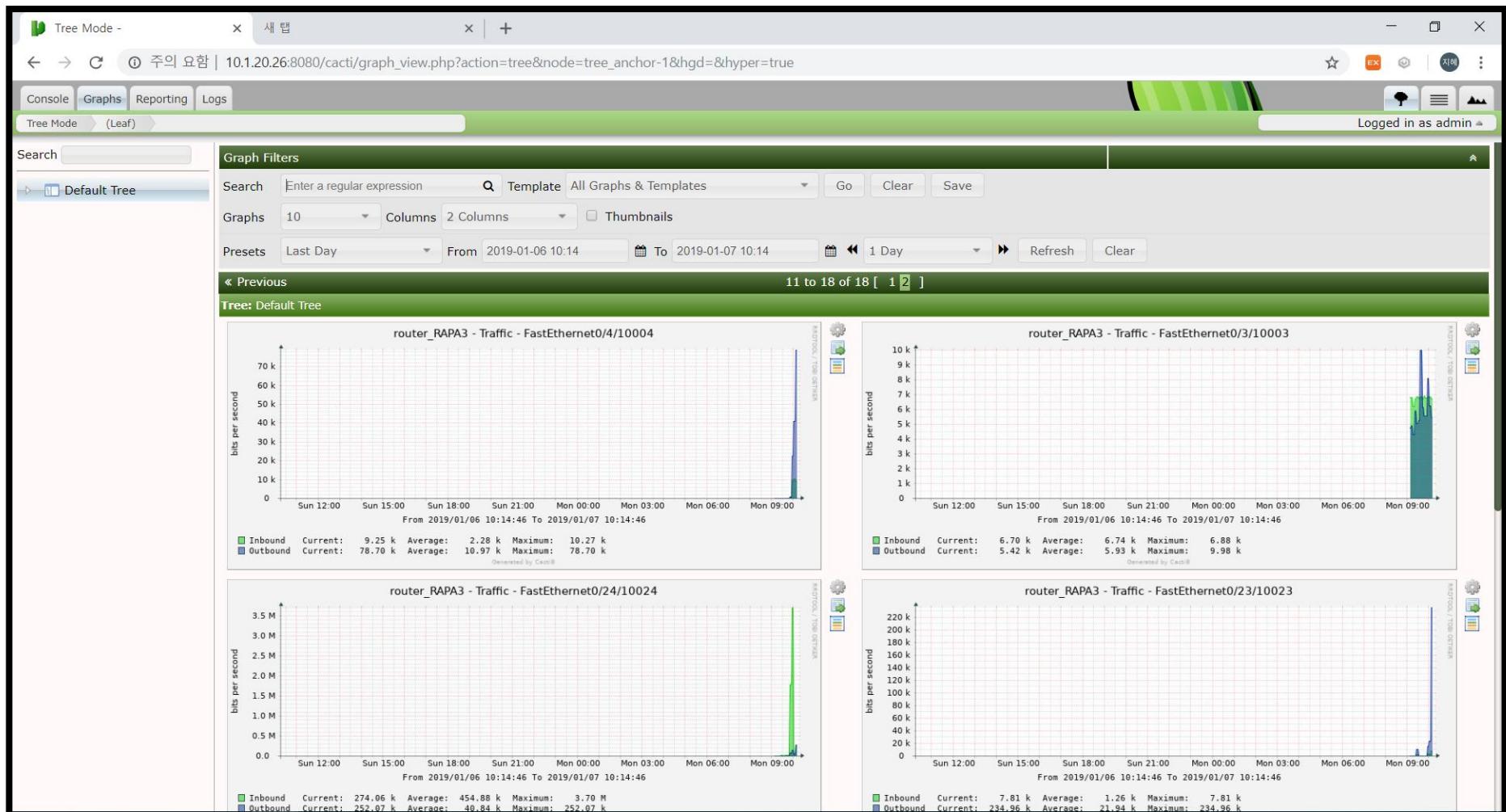
Cacti 실행(NMS 측정)

4) Cacti 측정결과(switch_RAPA3)



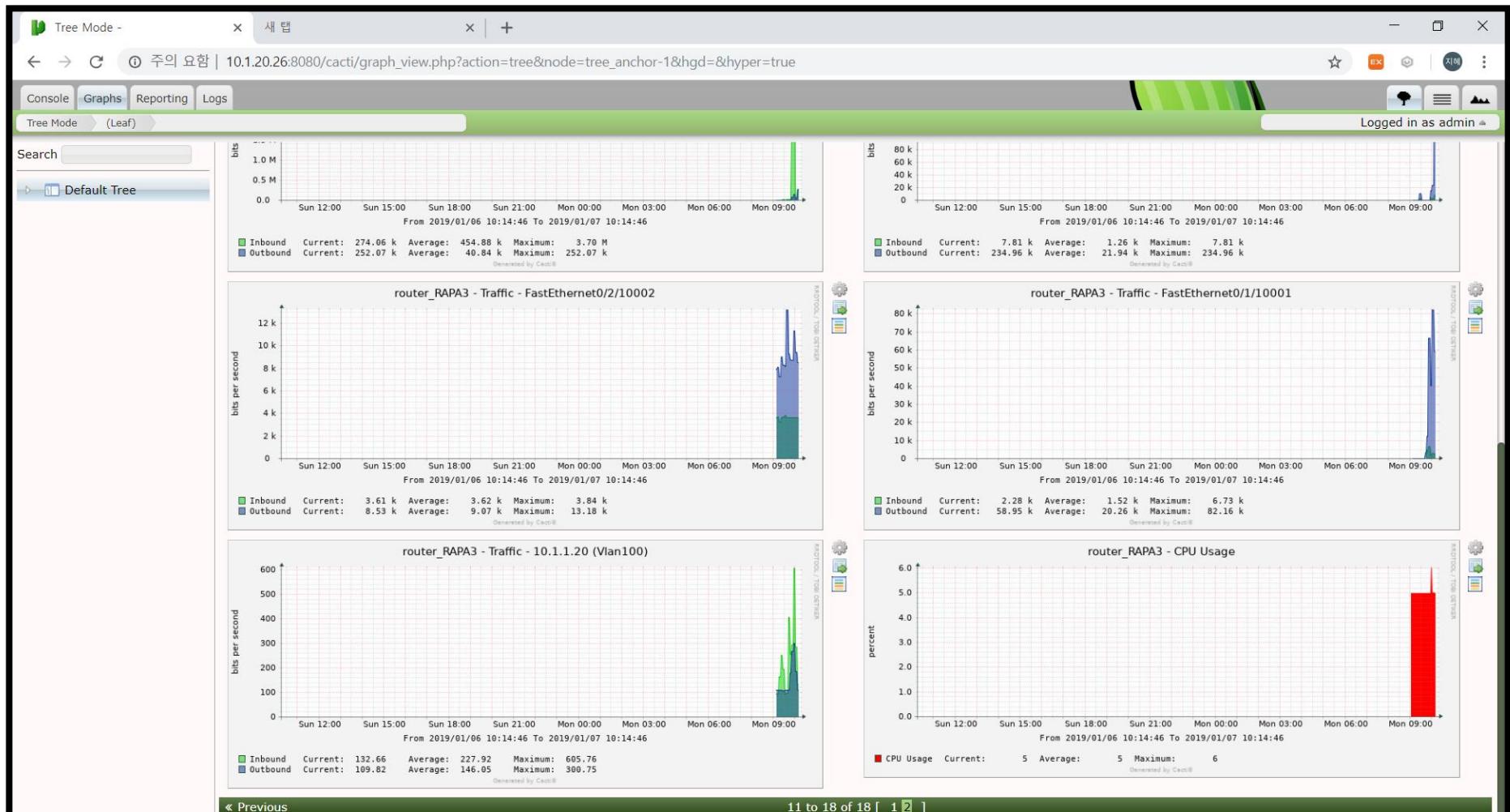
Cacti 실행(NMS 측정)

4) Cacti 측정결과(switch_RAPA3)



Cacti 실행(NMS 측정)

4) Cacti 측정결과(switch_RAPA3)



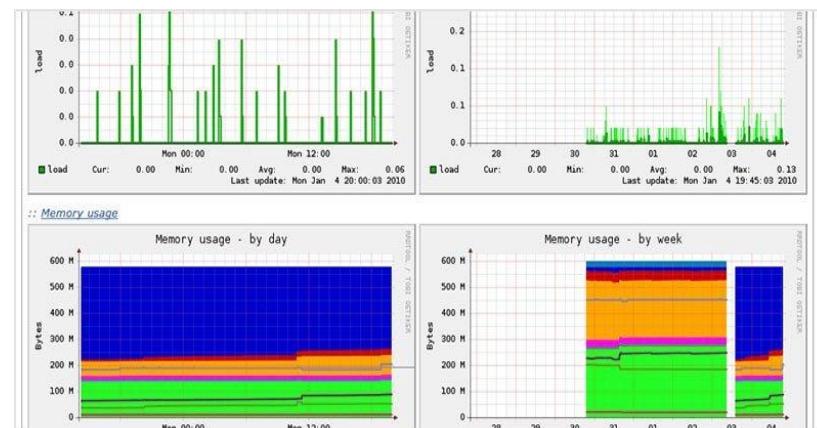
PRTG / JFFNMS 측정

PRTG 란?

- Paessler 사가 제공하는 시스템 모니터링도구로 NMS, SNMP로 시스템 사용량부터 세부적인 사항까지 모두 파악 가능하다. 센서 100개 까지는 무료이지만 초과는 라이선스 구입을 해야 한다. 필자는 지금까지 했었던 오픈스택과 KVM을 측정해보기 위해 이 도구를 사용하였다.

JFFNMS 란?

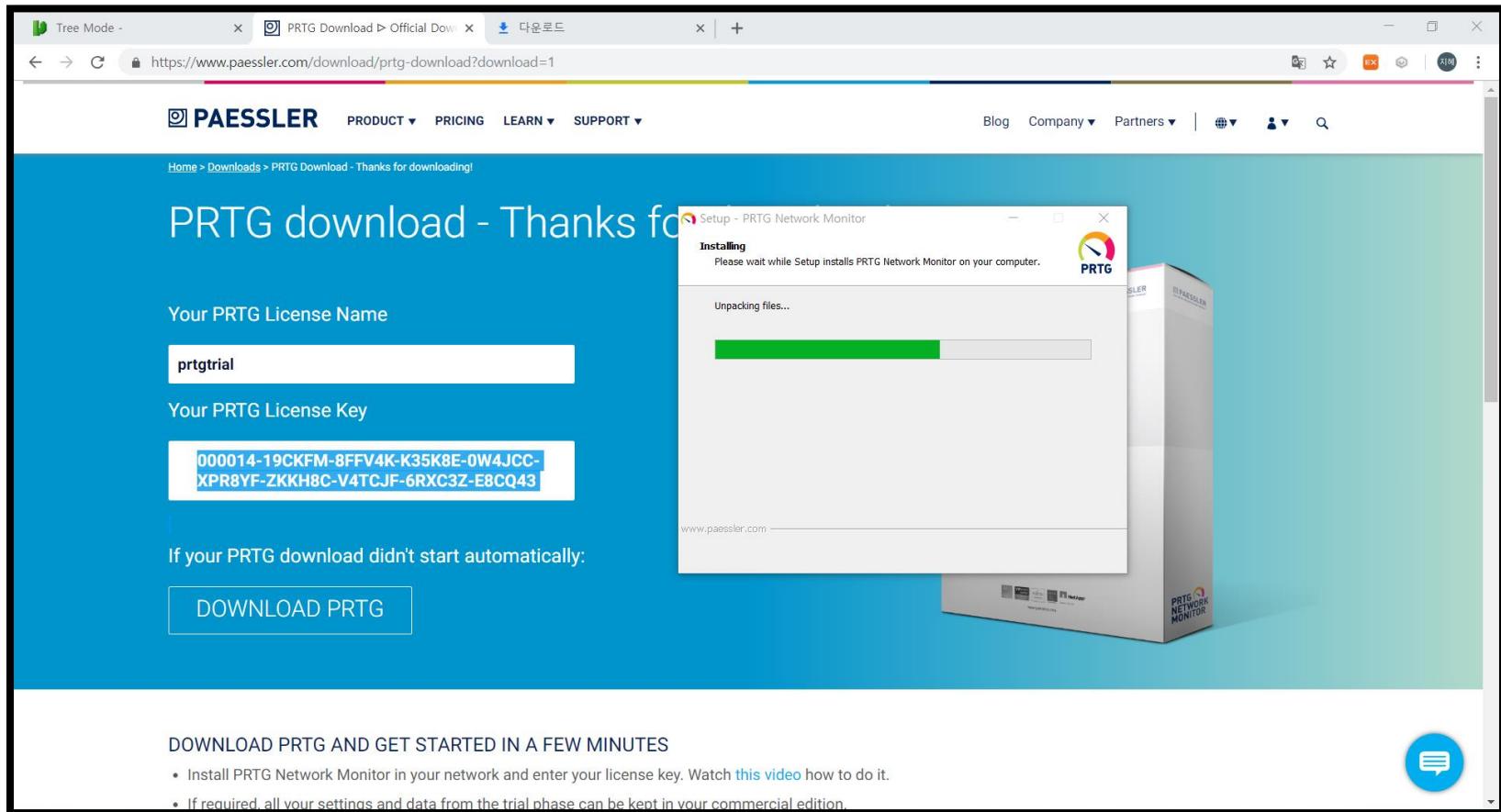
- 마찬가지로 네트워크 관리 및 감시 시스템으로 SNMP, NMS를 통해 다양한 호스트 및 프로토콜의 정보를 순간 순간마다 살펴볼 수 있다.



PRTG 실행(NMS 측정)

1) PRTG에 SNMP을려 측정해보기

- PRTG를 검색해서 다운받아 설치한다. Free라 License Key 값을 넣어주면 사용가능하다.
- 이전의 CACTI로 측정했던 그대로 스위치에 연결된 오픈스택, 그 외의 KVM도 같이 측정해볼 예정이다.



PRTG 실행(NMS 측정)

2) PRTG에 Sensor 추가하기

- 측정하고 싶은 switch의 ip를 입력하고 161번 포트(SNMP)를 열어준다. (마찬가지로 방화벽과 포트를 열어준다!)

Add Sensor Wizard

Device Selection
Please enter the device connection data

Device Name/Alias: main_switc
IP Address/DNS Name: 10.1.1.10
SNMP Community String: public (Standard is 'public')
SNMP Port: 161 (Standard is '161')

Help
Please enter a symbolic name for the device that you want to monitor "My Router" or "Mail Server".

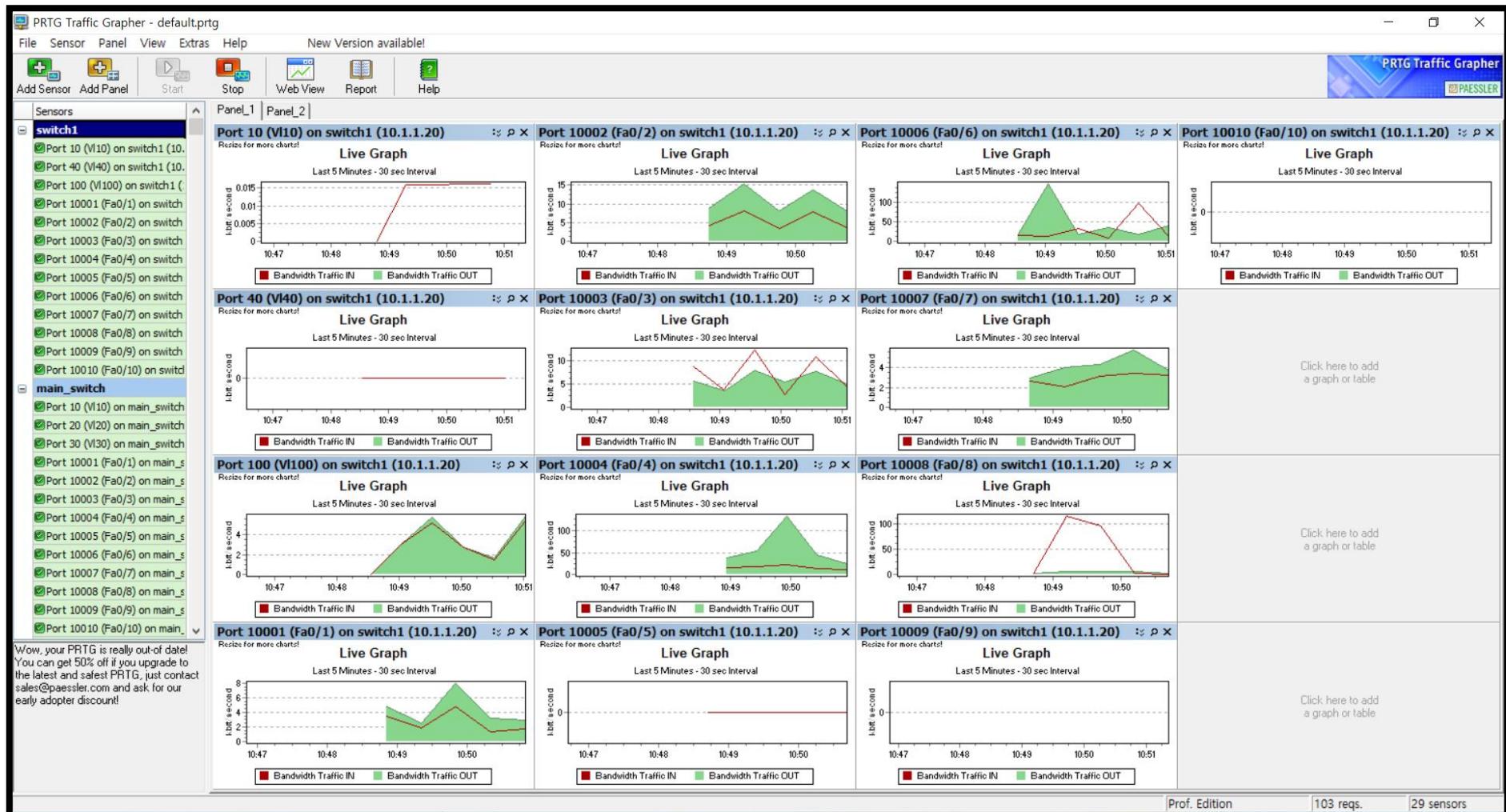
< Back Next >

Port 10 (V10) on switch1 (10.1.1.20) Live Graph
Port 10002 (Fa0/2) on switch1 (10.1.1.20) Live Graph
Port 10006 (Fa0/6) on switch1 (10.1.1.20) Live Graph
Port 40 (V140) on switch1 (10.1.1.20) Live Graph
Port 10003 (Fa0/3) on switch1 (10.1.1.20) Live Graph
Port 10007 (Fa0/7) on switch1 (10.1.1.20) Live Graph
Port 100 (V100) on switch1 (10.1.1.20) Live Graph
Port 10004 (Fa0/4) on switch1 (10.1.1.20) Live Graph
Port 10008 (Fa0/8) on switch1 (10.1.1.20) Live Graph
Port 10001 (Fa0/1) on switch1 (10.1.1.20) Live Graph
Port 10005 (Fa0/5) on switch1 (10.1.1.20) Live Graph
Port 10009 (Fa0/9) on switch1 (10.1.1.20) Live Graph

30초마다 사용량이 갱신된다!

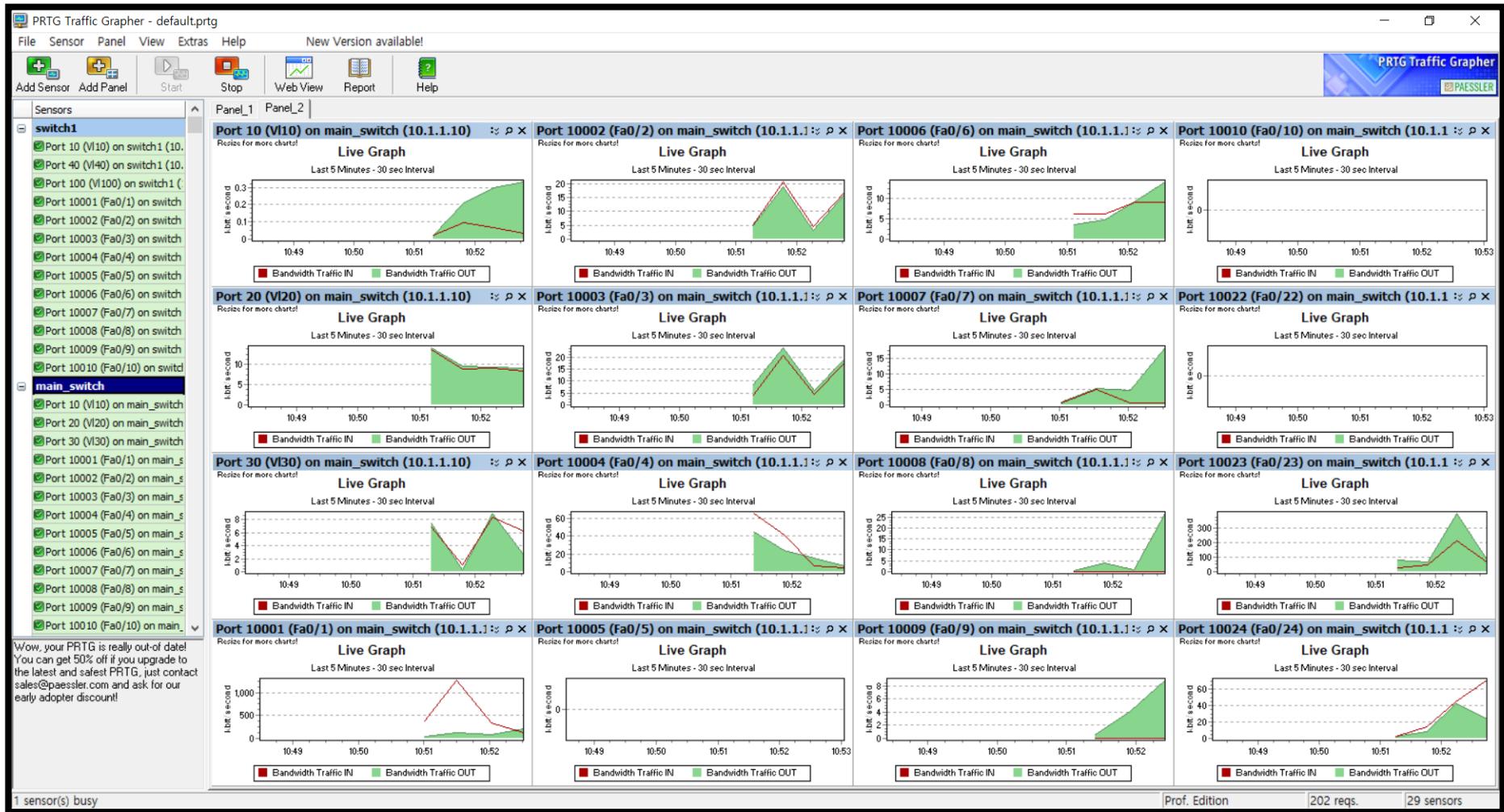
PRTG 실행(NMS 측정)

3) PRTG로 전체 사용량 측정(switch_RAPA3)



PRTG 실행(NMS 측정)

3) PRTG로 전체 사용량 측정(main_switch_RAPA1)



JFFNMS 실행(NMS측정)

1) JFFNMS를 가상머신에 설치, SNMP 올려 측정해보기

-JFFNMS의 경우, 미리 준비된 iso를 통해 VM에 올릴 수 있었다. 올린 VM의 IP를 치고 들어가면 해당 화면이 보인다.

➤ 이전의 PRTG로 측정했던 그대로 스위치에 연결된 오픈스택, 그 외의 KVM도 같이 측정해볼 예정이다.

The screenshot shows the JFFNMS 0.9.4 Start Page. At the top, there are tabs for 'Views: Start Page', 'Performance', and 'Administration'. On the left, there's a sidebar titled 'Menu' with options like 'Users and Customers', 'Hosts and Interfaces', 'Reports', 'Internal Configuration', and 'System Setup'. The main content area has sections for 'Statistics' (Alarms: All OK, Hosts: 4, Interfaces: 14), 'News' (JFFNMS 0.9.3 RC1 (01 Apr 2012), JFFNMS 0.9.1 Released (05 Jun 2011), JFFNMS 0.9.0 Released (21 May 2011), and JFFNMS version 0.9.0 at RC3 (04 May 2011)).

Statistics

- Alarms: All OK
- Hosts: 4
- Interfaces: 14

News

- JFFNMS 0.9.3 RC1** (01 Apr 2012)
The first release candidate for the open-source Network Management System JFFNMS has been uploaded to sourceforge. You can find it at <https://sourceforge.net/projects/jffnms/files/jffnms%20RC/>. This version fixes many bugs within the poller and greatly increases the poller rate.
- JFFNMS 0.9.1 Released** (05 Jun 2011)
The Network Management System JFFNMS has been updated to version 0.9.1. This version is a bugfix release only with no new added features. If you are running version 0.9.0 it is strongly recommended to upgrade to 0.9.1 Version 0.9 is planned to be a maintenance only stream which will have some minor feature enhancements but [...]
- JFFNMS 0.9.0 Released** (21 May 2011)
After 3 release candidates JFFNMS is now at version 0.9.0. Both the web frontend and backend (engines) have had extensive re-work done to them to cleanup and tighten up the code. There should be a lot less warnings and errors from PHP when you set to a higher error level. Fixed error in syslog consolidator [...]
- JFFNMS version 0.9.0 at RC3** (04 May 2011)
The latest Release Candidate for JFFNMS 0.9.0 has been released. I am expecting this to be the last RC before 0.9.0 comes out. It has significantly better job control between the parent and child processes for the engines and some clean-ups that I have previously missed. The files are found on SourceForge at <https://sourceforge.net/projects/jffnms/files/jffnms%20RC/>

JFFNMS 실행(NMS측정)

2) 측정하기 위한 host와 Zone 설정하기

- Administration > Hosts and Interfaces > Zones / Hosts

➤ RAPA3라고 Zone부터 만들어주고 그 Zone에 있는 rapa3_switch, kvm, kvm_CentOS, openstack을 만들어준다.

JFFNMS Administration - Zones

Action	ID	Zone	ShortName	Image	Visibility	Network Discovery Enabled	Network Discovery Seeds CIDR	Allow Private IPs	SNMP Communities CSV	Max Hops to scan	Re-Scan
Edit	3	rapa3	UNK	unknown.png	Show					1 - Just the specified Subnets	Every 24 Hours

JFFNMS Administration - Hosts

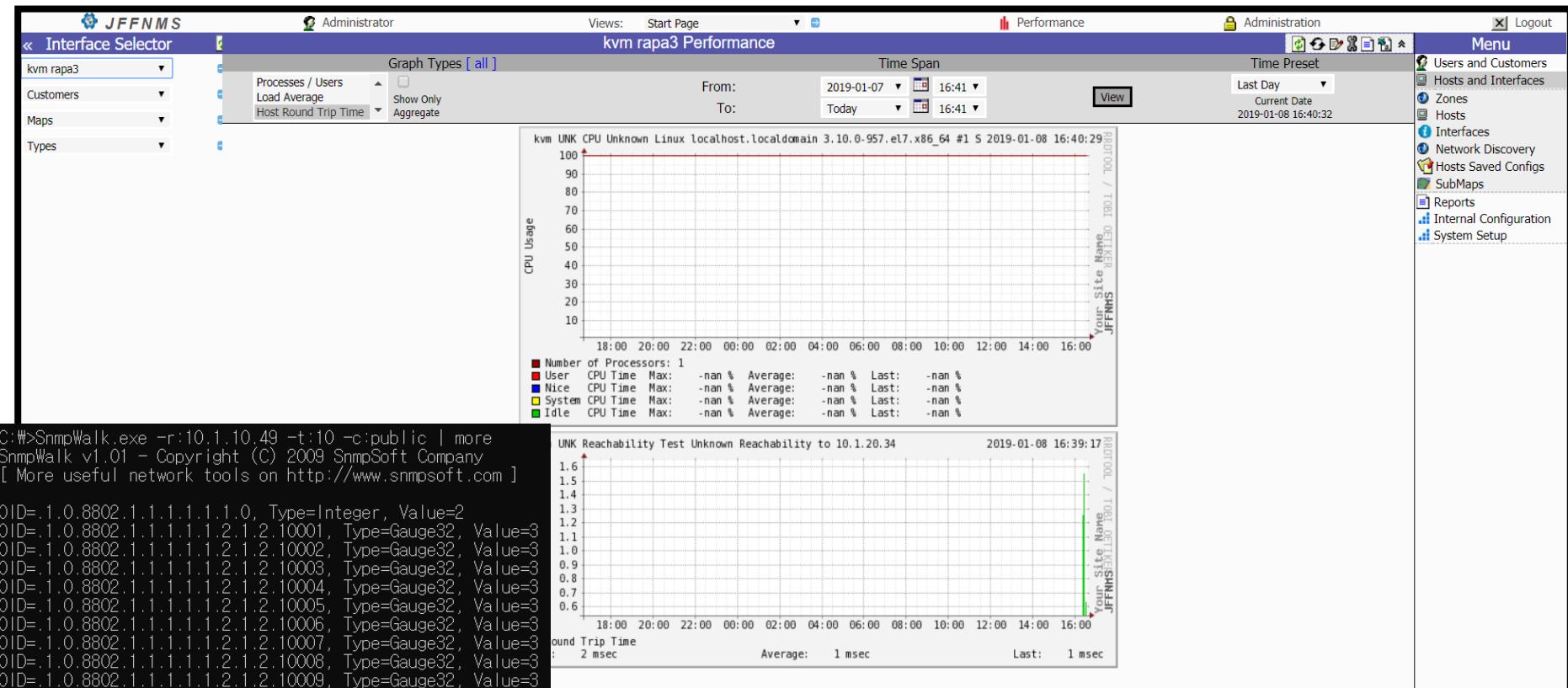
Action	ID	Name	Zone	IP Address	R/O Community	R/W Community	AutoDiscovery Policy	AD Default Customer	Visibility	Polling	Main Interface
Save	new	kvm_CentOS	rapa3	10.1.20.38	SNMPv2c	public	Not Set	No Autodiscovery	Unknown Customer	Show	

JFFNMS Administration - Hosts

Action	ID	Name	Zone	IP Address	R/O Community	R/W Community	AutoDiscovery Policy	AD Default Customer	Visibility	Polling	Main Interface(s)	TFTPd Server	IP	Config
Edit Host	11	kvm	rapa3	10.1.20.34	SNMPv2 Set	Not Set	No Autodiscovery	Unknown Customer	Show	<input checked="" type="checkbox"/>	(None Set)	No Co		
Edit Host	10	kvm_CentOS	rapa3	10.1.20.38	SNMPv2 Set	Not Set	No Autodiscovery	Unknown Customer	Show	<input checked="" type="checkbox"/>	(None Set)	No Co		
Edit Host	12	rapa3_switch	rapa3	10.1.1.20	SNMPv2 Set	Not Set	No Autodiscovery	Unknown Customer	Show	<input checked="" type="checkbox"/>	(None Set)	No Co		

JFFNMS 실행(NMS측정)

2) JFFNMS로 KVM 측정



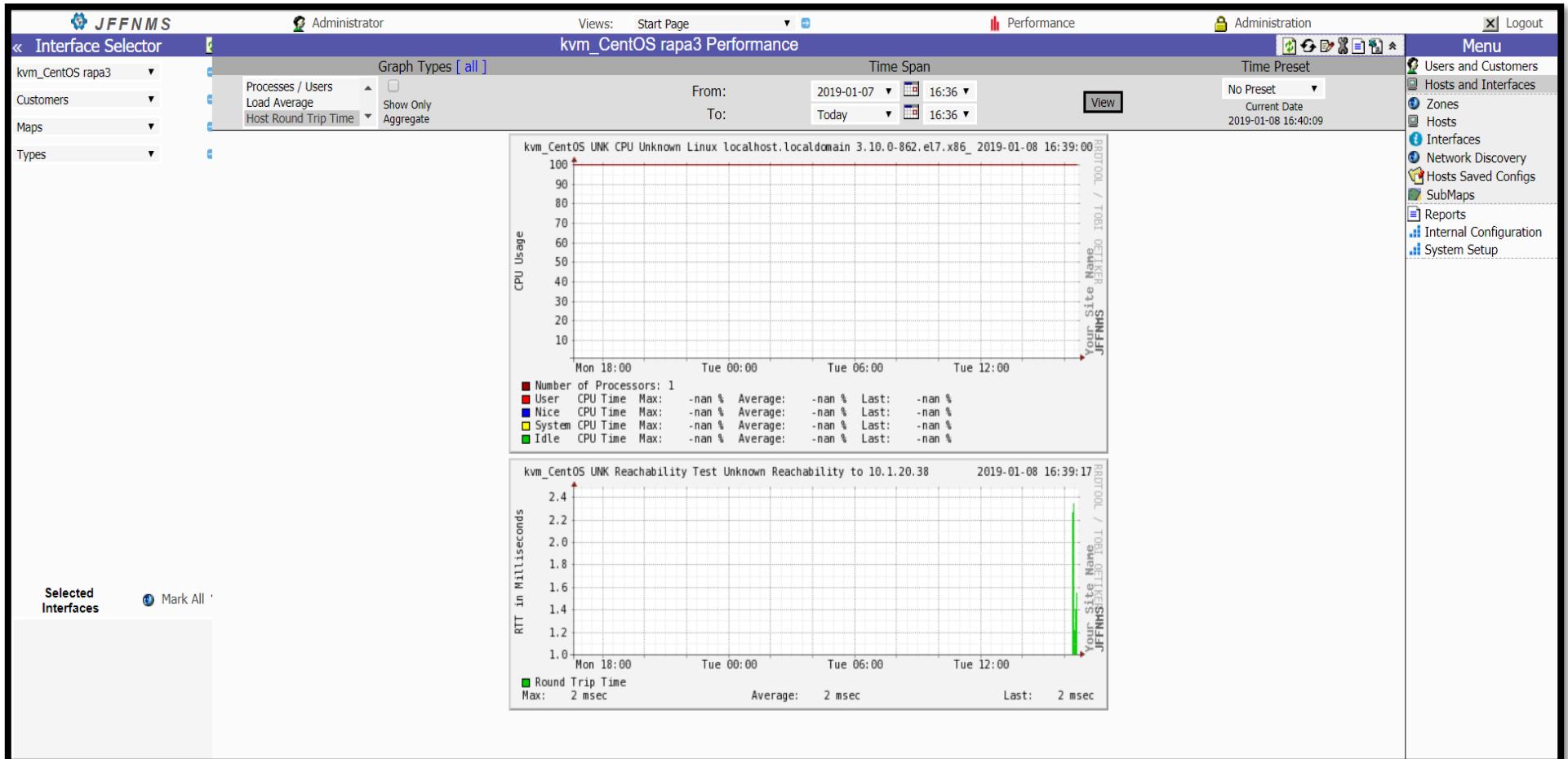
만약 SNMP가 열렸는지 확인하고 싶다면 SnmpWalk를 통해 MIB확인하자!

CMD명령어

C: W> SnmpWalk.exe -r : [확인하려는 IP] -t: 10 -c: public

JFFNMS 실행(NMS측정)

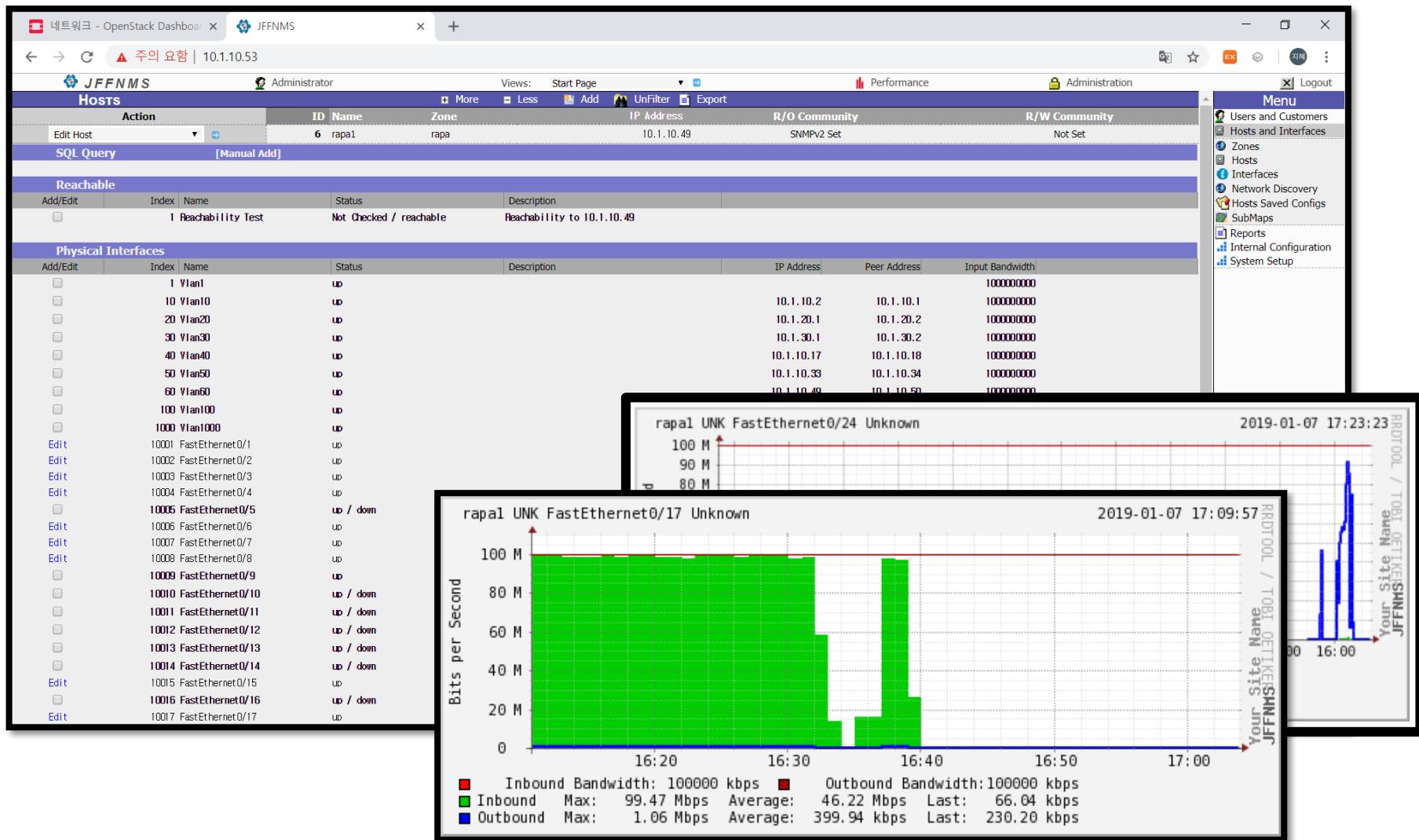
2) JFFNMS로 KVM_CentOS 측정



KVM에 올린 CentOS 또한 측정가능하다!

JFFNMS 실행(NMS측정)

3) JFFNMS로 RAPA1스위치 측정



JFFNMS 실행(NMS측정)

4) JFFNMS로 Openstack 측정

The screenshot shows the JFFNMS interface for monitoring an OpenStack host. The main window displays various system components and their statuses.

Hosts:

Action	ID	Name	Zone	IP Address	R/O Community	R/W Community
Edit Host	13	openstack10_5	rapa	10.1.10.5	SNMPv2 Set	Not Set

Storage:

Add/Edit	Index	Name	Status	Disk Type	Size (Bytes)	Description
	1	Physical memory	up	RAM	813332932	
	3	Virtual memory	up	VirtualMemory	16454820032	
	6	Memory buffers	up	Other	813332932	
	7	Cached memory	up	Other	1143631872	
	8	Shared memory	up	Other	269668352	
	10	Swap space	up	VirtualMemory	8321495040	
	31	/	up	FixedDisk	53660876800	
	36	/dev/shm	up	FixedDisk	4066664448	
	38	/run	up	FixedDisk	4066664448	
	39	/sys/fs/cgroup	up	FixedDisk	4066664448	
	57	/srv/node/swiftloopback	up	FixedDisk	1945976832	
	58	/boot	up	FixedDisk	1063256064	
	59	/home	up	FixedDisk	1064623411200	
	61	/run/user/0	up	FixedDisk	813334528	

SQL Query: [Manual Add]

Reachable:

Add/Edit	Index	Name	Status	Description
	1	Reachability Test	Not Checked / reachable	Reachability to 10.1.10.5

Physical Interfaces:

Add/Edit	Index	Name	Status	Description	IP Address	Peer Address	Input Bandwidth
	1	lo	up		127.0.0.1	127.0.0.2	1000000
	2	enp5s0	up		10.1.10.5	10.1.10.6	10000000
	3	ovs-system	down				

Services: (Listed under Physical Interfaces)

1	broket	running	Application	12
2	chronyd	running	Application /usr/sbin/chronyd	1
3	cinder-api	running	Application /usr/bin/python2	5
4	cinder-backup	running	Application /usr/bin/python2	1
5	cinder-schedule	running	Application /usr/bin/python2	1
6	cinder-volume	running	Application /usr/bin/python2	2
7	crond	running	Application /usr/sbin/crond	1
8	crypto	running	Application	1
9	dbus-daemon	running	Application /usr/bin/dbus-daem	1

감사합니다