

2020 개인 프로젝트
OpenStack 아키텍처 분석

OpenStack
각 서비스별 아키텍처 구현& 분석
트러블 슈팅


이름
이정근

이메일
mung0001@naver.com


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

개 정 이 력

개정 번호	개정 내용 요약	추가/수정 항목	개정 일자
1.0	최초 제정 승인 (OpenStack Roadmap)	시작	2020.09.05
2.0	추가 수정 승인 (Keystone)	Keystone 항목추가	2020.09.12
2.1	추가 수정 승인 (Glance)	Glance 항목추가	2020.09.19
2.2	추가 수정 승인 (Nova)	Nova 항목추가	2020.09.26
2.3	추가 수정 승인 (Neutron)	Neutron 항목추가	2020.10.03
2.4	추가 수정 승인 (Cinder)	Cinder 항목추가	2020.10.10
2.5	추가 수정 승인 (Ceilometer)	Ceilometer 항목추가	2020.10.17
2.6	추가 수정 승인 (Horizon)	Horizon 항목추가	2020.10.24
2.7	추가 수정 승인 (Swift)	Swift 항목추가	2020.10.31
2.8	추가 수정 승인 (Heat)	Heat 항목추가	2020.11.7
3.0	최종 수정 승인	OpnStack BigTent 최종 수정	2020.11.14


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

문 서 규 칙

	Report	Version	Last Modified	④
	①	②	③	


- 작성은 Microsoft Word Office 365(v.1908)으로 작성되었으며, 저장시 PDF 형식으로 저장한다.
- 확인은 Adobe Arcrobat Reader로 사용한다.
- Report(①)에는 핵심 개정 내용을 표기한다.
- Verion (②)에는 문서의 개정 번호를 표기한다.
- Last Modified(③)에는 문서의 개정 일자를 표기한다.
- ④ 에는 프로젝트 명을 표기한다.
- OpenStack 설치 시 하단의 표와 같이 Node 이름에 맞추어 설치를 진행한다.

구문	의미
\$ all>	모든 Node에 설치를 의미
\$ controller>	controller Node에만 설치를 의미
\$ controller(keystone)>	keystone으로 인증받은 controller를 의미
\$ compute& network>	compute, network Node를 의미
\$ all(Storage)>	모든 Storage Node를 의미


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

목차


개 정 이 력	1
문 서 규 칙	2
PROJECT 개요.....	7
Project 주제	7
OPENSTACK 개요.....	7
OpenStack이란?.....	7
OpenStack과 Cloud Computing	7
System, Network 기본	9
OpenStack Roadmap.....	11
OpenStack Node 별 서비스 설치 구성도	13
OPENSTACK 기본설정	14
OpenStack Architecture.....	14
OpenStack Ussuri repository 등록	15
OpenStack NTP(Network Time Protocol) 설치	15
OpenStack DB(MariaDB) 설치.....	16
OpenStack Queue(RabbitMQ) 설치	16
OpenStack Cache(Memcached) 설치	17
인증을 관리하는 서비스 : KEYSTONE.....	18
Keystone의 구성요소	18
Keystone의 논리 아키텍처.....	20
Keystone의 OpenStack 에서의 역할	20
Keystone Install.....	21
이미지를 관리하는 서비스 : GLANCE.....	23
Glance의 구성요소	24

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


Glance의 논리 아키텍처	24
Glance의 Image	25
Glance Install	25
 가상의 서버를 생성하는 서비스 : NOVA	 30
Nova의 고려요소	30
Nova의 논리 아키텍처 및 구성요소	31
Nova 하이퍼바이저의 종류	32
Nova Install	32
 네트워크를 관리하는 서비스 : NEUTRON	 40
Neutron의 논리 아키텍처 및 구성요소	41
Neutron의 네트워킹 프로세스	42
Neutron network의 종류	43
Neutron과 VRRP, DVR	44
Neutron Install	44
 블록 스토리지 서비스 : CINDER	 63
Cinder의 구성요소	64
Cinder의 논리 아키텍처	65
Cinder driver type	66
Cinder의 Install	66
 리소스의 사용량과 부하를 관리하는 서비스 : CEILOMETER	 71
Ceilometer의 구성요소	72
Ceilometer의 데이터 수집	73
Ceilometer의 데이터 처리	73
Ceilometer의 데이터 요청	74
Ceilometer의 데이터 처리 및 변형	75
Ceilometer의 아키텍처	76
Ceilometer Install	77
 외부 인터페이스 대시보드 서비스 : HORIZON	 77

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Horizon의 아키텍처	77
Horizon Install	78
 오브젝트 스토리지 관리 서비스 : SWIFT	 79
Swift의 구성요소	79
Swift의 아키텍처	80
Swift Ring.....	81
Swift의 데이터 관리.....	82
Swift와 Keystone	82
Swift의 이레이저 코딩기능과 스토리지 정책	84
Swift Install	84
 오케스트레이션 서비스 : HEAT	 91
Heat의 구성요소와 논리 아키텍처.....	92
Heat Install.....	93
 데이터베이스 서비스 : TROVE	 99
차후 업데이트 예정입니다.....	99
 데이터 프로세싱 서비스 : SAHARA.....	 99
차후 업데이트 예정입니다.....	99
 베어메탈 서비스 : IRONIC.....	 99
차후 업데이트 예정입니다.....	100
 각 서비스별 트러블 슈팅	 100
Keystone.....	100
Glance	101
Nova	101
Neutron	102
Cinder.....	104
Swift	105
Heat.....	105

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Horizon 106

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Project 개요

Project 주제

- OpenStack 서비스의 오픈소스를 두루 다뤄보고 아키텍처를 분석하는 능력을 함양
- OpenStack 내에서 각 서비스들이 어떠한 역할을 수행하며, 이를 어떻게 구현하고, 응용할 수 있는지에 대한 학습
- OpenStack의 각 서비스를 구현 및 다뤄보며 CloudComputing의 대한 개념을 학습
- OpenStack 및 AWS(Amazon Web Service)와의 다른 점을 학습하며 일반적인 Private Cloud, Public Cloud에 대한 개념을 학습


OpenStack 개요

OpenStack이란?

- 오픈스택이란 컴퓨트, 오브젝트 스토리지, 이미지, 인증 서비스 등의 오픈소스를 활용해 유기적으로 연결되어 하나의 커다란 클라우드 컴퓨팅 시스템을 구축하는 것을 의미한다.

OpenStack과 Cloud Computing

- **Cloud Computing**
 - 클라우드 컴퓨팅이란 인터넷이 가능한 디바이스(스마트폰, 스마트패드, 스마트 TV 등)으로 클라우드에서 데이터를 처리하며, 저장 및 관리하는 컴퓨팅 시스템을 의미한다.
- **Cloud Service 종류**


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- IaaS(Infrastructure as a Service): 서버, 스토리지, 네트워크를 가상화 환경으로 만들어 필요에 따라 인프라 자원을 제공하는 서비스
- PaaS(Platform as a Service): 웹에서 개발 플랫폼을 제공하는 서비스
- SaaS(Desktop as a Service): 클라우드 인프라를 이용해 os가 설치된 인스턴스를 제공하는 서비스
- BaaS(Backend as a Service): 모바일 환경에 맞춰 구현하기 힘든 백엔드 부분을 제공하는 서비스
- Public Cloud: 언제든지 접근이 가능한 클라우드 서비스
- Private Cloud: 외부에서는 접근이 불가능한 사내 클라우드 서비스

- **Cloud Server의 핵심 Compute, Storage**
 - Compute Service: 사용자가 원하는 운영체제가 탑재된 컴퓨터나 서버를 인터넷에서 사용할 수 있게 제공하는 서비스
 - Storage Service: 사용자가 소유한 데이터나 음악, 동영상, 문서 파일을 인터넷에 있는 스토리지에 저장, 삭제, 공유할 수 있는 서비스

- **Hypervisor**
 - Hypervisor: 가상 머신 모니터라고도 하며, 호스트 컴퓨터 한 대에서 운영체제 다수를 동시에 실행하는 논리적인 플랫폼을 의미
 - Hypervisor은 하드웨어에 직접 설치해서 실행되는 Native, 베어메탈 방식과 애플리케이션처럼 프로그램으로 설치되는 Hosted 방식이 존재
 - 가상화 방식에 따라 하이퍼바이저가 분류되며, 하드웨어를 모두 가상화하는 방식으로 게스트 운영체제를 변경하지 않고, 다양한 운영체제로 이용할 수 있는 Native 방식과 하이퍼바이저로만 제어가 가능한 방식으로, 높은 성능의 유지가 가능하지만, 오픈 소스가 아니면 운영이 불가능한 특징을 가지고 이는 반가상화 방식이 존재

- **Hypervisor의 종류**

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- KVM: 오픈스택의 기본 하이퍼바이저로 전가상하 방식을 지원(inter VT or AMD-V)
- Xen: Center을 이용한 관리 기능, 스토리지 지원과 실시간 마이그레이션,고가용성 기능처럼 데이터센터에서 요구하는 확장 기능을 제공
- Hyper-V: 디바이스 드라이버가 부모 파티션 이에서 동작하며, 콘솔 OS의 역할을 부모 파티션이 수행
- VMware vSphere ESX: 작은 하드웨어에서도 애플리케이션을 통합할 수 있도록 서버를 가상화해주는 무료 베어메탈 하이퍼바이저
- Docker: 리눅스 기반의 컨테이너 런타임 오픈 소스로, 가상 머신과 기능이 유사하며, 가상 머신보다 가벼운 형태(프로세스)로 배포가 가능
- VirtualBox: 리눅스, OS, X, 솔라리스, 윈도우를 게스트 운영체제로 가상화하는 x86 가상화 소프트웨어
- VMware Workstation: 게스트 운영체제에 설치할 수 있는 드라이버 및 기타 소프트웨어의 묶음
- Parallels Dekstop: 맥용 인텔 프로세서가 있는 매킨토시 컴퓨터에 하드웨어 가상화를 제공하려 만든 소프트웨어


System, Network 기본

- 고정 IP, 유동 IP

- 고정 IP (Fixed IP) : 인터넷 공유기를 연결해 고정으로 할당받는 IP
- 유동 IP (Floating IP) : 가상 인스턴스가 외부에서 접근할 수 있도록 할당하는 인터넷이 가능한 Public IP

- 클래스의 범위

- A 클래스 : 1 ~ 126
- B 클래스 : 128 ~ 191


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- C 클래스 : 192 ~ 223
- D 클래스 : 224 ~ 230 (멀티캐스트용의 예약 클래스)
- E 클래스 : 240 ~ 254 (연구 개발 목적의 예약 클래스)

- **CIDR(Classless Inter-Domain Routing)**
 - 클래스가 없는 도메인간 라우팅 기법으로 기존 IP 할당 방식인 네트워크 클래스를 대체하며, 급격히 부족해지는 IPv4 주소를 보다 효율적으로 사용하기 위해 사용
 - ex) 192.168.1.0/24

- **오픈플로(OpenFlow)**
 - SDN의 근간이 되는 기술로 SDN 아키텍처의 컨트롤 레이어와 인프라스트럭처 레이어 사이에 정의된 최초의 표준 통신 인터페이스를 의미
 - 흐름정보로 패킷의 전달 경로와 방식을 제어
 - 오픈 플로는 오픈플러 컨트롤러와 오픈플로로 지원하는 네트워크 장비(라우터, 스위치) 사이에서 커뮤니케이션 역할을 담당


- **네트워크 장비**
 - 라우터 : 인터넷 등 서로 다른 네트워크를 연결할 때 사용하는 장비로, 데이터 패킷이 목적지까지 갈 수 있는 경로를 검사하여 최적의 경로를 탐색하는 것을 라우팅이라 하며, 경로가 결정되면 결정된 길로 데이터를 패킷하는 것을 스위칭이라 칭함
 - 허브 : 인터넷이 등장하기 이전, 컴퓨터와 컴퓨터를 연결해 네트워크를 구성하는 장비로 멀티포트(Multitport) 또는 리피터(Repeater)이라 할 수 있음
 - CSMA/CD(Carrier Sense Multiple Access/Collision Detect) : 이더넷 전송 프로토콜을 IEEE 802.3 표준에 규격화 되어 있음
 - 브릿지 : 콜리전 도메인을 나누어 서로 통신이 가능하도록 다리처럼 연결해 주는 네트워크 장비

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- 스위치 : 브리지와 역할이 동일하지만, 소프트웨어적으로 처리하는 스위치가 소프트웨어적으로 처리하는 브리지 보다 속도가 더 빠름
- **블록 스토리지와 오브젝트 스토리지**
 - 블록 스토리지 : 컴퓨터의 용량을 추가하는 것처럼 클라우드 상의 하드 디스크를 블록 스토리지라 칭함
 - 오브젝트 스토리지 : 사용자 계정별로 저장 공간을 할당할 수 있는 스토리지 시스템으로 블록 스토리지와는 다르게 단독으로 구성이 가능하며, 계정의 컨테이너 파일이나 데이터를 저장할 수 있는 저장 공간을 의미

OpenStack Roadmap

- 사용 툴 : VMwareworkstation Pro16
 - OpenStack Version : Ussuri
 - OpenStack Install order
1. 기본설정
 2. Keystone
 3. Glance
 4. Nova
 5. Neutron
 6. Cinder(6번까지가 기본 OpenStack 설치순서)
 7. Horizon(7번 부터는 개인의 필요에 따라 설치)
 8. Swift
 9. Heat
 10. Gnocch

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

11. Trove


12. Designate

13. Barbican

14. Rally

15. Manila


OS	HOSTNAME	CPU/Therad	RAM	DISK	Network-interface
CentOS8 (core)	controller	2/4	4096	100G	NAT (192.168.10.10)
	network	2/4	2048	40G	NAT (192.168.10.20)
	compute	2/4	4096	40G	NAT (192.168.10.30)
	storage1	1/2	1024	50G	NAT (192.168.10.50)
	storage2	1/2	1024	50G	NAT (192.168.10.51)
	storage3	1/2	1024	50G	NAT (192.168.10.52)

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

OpenStack Node 별 서비스 설치 구성도

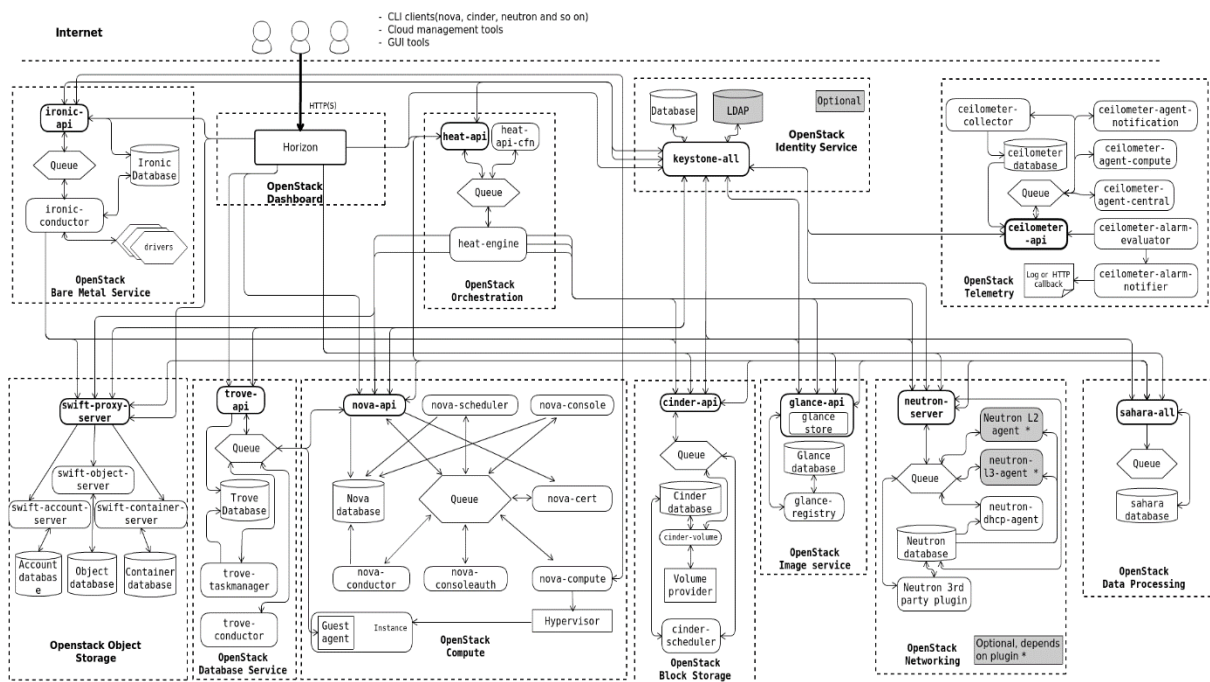
-----		-----		-----	
[Controller Node]		[Compute Node]		[Network Node]	
		Libvirt		Open vSwitch	
MariaDB RabbitMQ		Nova compute		L2 Agent	
Memcached Keystone		Open vSwitch		L3 Agent	
httpd Cinder API		L2 Agent		metadata agent	
Nova-API Compute		Cinder-LVM		Swift-proxy	
L2 agent L3 agent		NFS		Heat API	
metadata agent		-----		API-CFN	
Neutron Server				Heat Engine	
-----				-----	

[Storage Node 1, 2, 3]					
Swift-account-auditor					
Swift-account-replicator					
Swift-account					
Swift-container-auditor					
Swift-container-replicator					
Swift-container-updater					
Swift-container					
Swift-object-auditor					
Swift-object-replicator					
Swift-object-updater					
Swift-swift-object					


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

OpenStack 기본설정

OpenStack Architecture



- 상단의 Architecture 그림과 같이 OpenStack은 다양한 오픈소스들을 사용하는 거대한 오픈소스로 구성되어져 있으며, 이에 따라 각 HOST 혹은 각 서비스들간의 소통은 위해 queue(RabbitMQ), Cache(Memcached), 그리고 각 설정 값들을 저장하기 위한 DB(MariaDB)가 필요로 한다.
- 이에 먼저 OpenStack의 각 서비스를 설치하기 전에 서로 간의 통신이 가능하도록 기본 설정이 필요하다.
- 기본적인 networkinterface, hostname, hosts 설정은 위의 표를 참고하도록 한다.

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

OpenStack Ussuri repository 등록

- OpenStack은 기본적으로 repository가 등록되어있지 않아, 이를 설치하기 위해서는 레포지터리에 등록이 필요하다.

```
$ all> dnf -y install centos-release-openstack-ussuri
# centos openstack-ussuri 버전을 릴리즈한다.

$ all> sed -i -e "s/enabled=1/enabled=0/g" /etc/yum.repos.d/CentOS-OpenStack-ussuri.repo
# ussuri.repo 를 레포지터리에 등록한다.

$ all> dnf --enablerepo=centos-openstack-ussuri -y upgrade
# ussuri 레포지터리의 업데이트를 진행한다.
```

OpenStack NTP(Network Time Protocol) 설치

```
$ all> dnf --enablerepo=centos-openstack-ussuri -y install openstack-selinux
# SELinux 를 설치한다.


$ all> dnf install -y wget
# wget 을 설치한다.

$ all> dnf --enablerepo=PowerTools -y install epel-release
# epel 레포지터리를 등록한다.

$ all> dnf --enablerepo=PowerTools -y install checkpolicy
# 만약 Checkpolicy 가 설치되어 있지 않으면, 패키지를 다운 받는다.

$ all> dnf -y install chrony
$ all> vi /etc/chrony.conf
# pool 2.centos.pool.ntp.org iburst → pool controller
# allow 10.10.10.0/24 → allow 192.168.10.0/24

$ all> systemctl enable --now chronyd
$ all> systemctl restart -noew chronyd
$ all> firewall-cmd --add-service=ntp --permanent
$ all> firewall-cmd -reload
# 서비스를 등록 및 재시작하고 방화벽에 등록
$ all> chronyc sources
210 Number of sources = 1
MS Name/IP address          Stratum Poll Reach LastRx Last sample
=====
^? controller                0    7    0    -    +0ns[  +0ns] +/-
0ns
# 확인이 가능하다.
```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

OpenStack DB(MariaDB) 설치

```
$ controller> dnf module -y install mariadb:10.3
$ controller> vi /etc/my.cnf.d/charaset.cnf
[mysqld]
character-set-server = utf8mb4
[client]
default-character-set = utf8mb4
# mariadb 를 설치 후, charaset 설정을 변경하기 위해 파일을 수정
# charset 은 사용하는 언어의 집합을 의미 (기본값 설정이 한글사용불가능)

$ controller> systemctl restart --now mariadb
$ controller> systemctl enable --now mariadb
# DB 를 재시작한다.

$ controller> firewall-cmd --add-service=mysql --permanent
$ controller> firewall-cmd --reload
# 방화벽을 설정한다.

$ controller> mysql_secure_installation
$ controller> mysql -u root -p
# 설정을 초기화 후, 비밀번호를 생성
```

OpenStack Queue(RabbitMQ) 설치


- RabbitMQ는 오픈소스 메시지 브로커 소프트웨어로 AMQP를 구현하는 역할을 수행하며, OpenStack 내부에서는 서로간의 통신을 위해 사용된다.

```
$ controller> dnf --enablerepo=PowerTools -y install rabbitmq-server
$ controller> vi /etc/my.cnf.d/mariadb-server.cnf
[mysqld]
.....
.....
max_connections=500
# 인증허용 시간 값을 추가

$ controller> systemctl restart mariadb rabbitmq-server
$ controller> systemctl enable mariadb rabbitmq-server
# RabbitMQ 서비스를 등록

$ controller> rabbitmqctl add_user openstack qwer1234
$ controller> rabbitmqctl set_permissions openstack ".*" ".*" ".*"
# rabbitmq를 사용할 openstack 유저를 패스워드 qwer1234로 생성하고 모든 권한을 부여

$ controller> vi rabbitmqctl.te
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

module rabbitmqctl 1.0;

require {
    type rabbitmq_t;
    type rabbitmq_var_log_t;
    type rabbitmq_var_lib_t;
    type etc_t;
    type init_t;
    class file write;
    class file getattr;
}

#===== rabbitmq_t =====
allow rabbitmq_t etc_t:file write;

#===== init_t =====
allow init_t rabbitmq_var_lib_t:file getattr;
allow init_t rabbitmq_var_log_t:file getattr;

$ controller> checkmodule -m -M -o rabbitmqctl.mod rabbitmqctl.te
$ controller> semodule_package --outfile rabbitmqctl.pp --module
rabbitmqctl.mod
$ controller> semodule -i rabbitmqctl.pp
# rabbitmq의 설정을 추가 후 등록시킵니다.

$ controller> firewall-cmd --add-port=5672/tcp --permanent
$ controller> firewall-cmd --reload
# 방화벽을 설정

```

OpenStack Cache(Memcached) 설치

- Memcached는 범용 분산 캐시 시스템으로, OpenStack 내부에서 캐시 값을 관리하는 역할을 수행한다.

```


$ controller> dnf --enablerepo=PowerTools -y install memcached

$ controller> vi /etc/sysconfig/memcached
OPTIONS="-l 0.0.0.0,::"
# 모두가 사용할 수 있도록 값을 수정

$ controller> systemctl restart memcached
$ controller> systemctl enable memcached
# Memcached 서비스를 등록

$ controller> firewall-cmd --add-service=memcache --permanent
$ controller> firewall-cmd --reload
# 방화벽을 설정

```

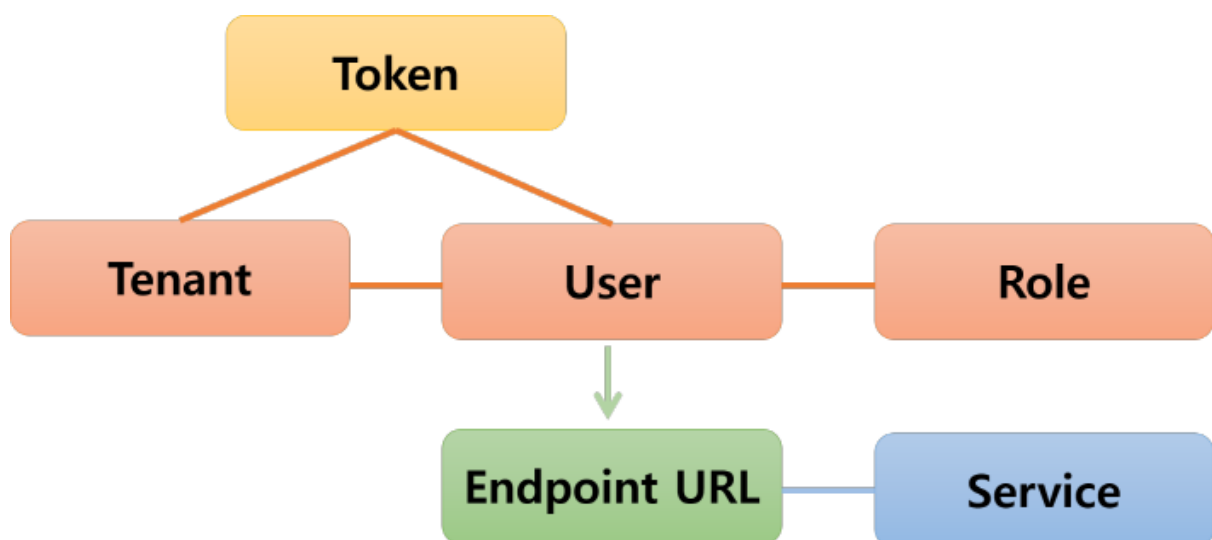
	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- OpenStack에서의 기본적인 환경설정은 서로 간의 통신과 시간을 동기화하기 위하여 NTP, DB, Cache를 사용한다는 것을 확인 할 수 있다.


인증을 관리하는 서비스 : Keystone

- Keystone은 인증 토큰을 비롯해 테넌트 및 사용자 관리, 서비스의 엔드포인트 URL 등을 관리하는 서비스를 의미한다.
- Keystone은 OpenStack의 백엔드에서 RBAD(Role Based access Control)을 통해 사용자의 접근을 제어하는 등의 인증(Identify) 서비스로 사용된다.

Keystone의 구성요소




구성요소	역할
User	사람 또는 오픈 스택 서비스를 이용하는 서비스 (nova, neutron, cinder 등)을 의미

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

	User은 특정 프로젝트에 할당할 수 있으며, 중복을 허용하지 않음
Authentication	사용자의 신분을 확인하는 절차로, 특정 값을 통해 Keystone이 이를 검증
	보통 인증을 위한 자료로는 ID, PW가 사용되며 Keystone은 인증 확인 시 인증토큰을 발행
Token	RBAD의 신분을 증명하기 위해 사용되는 텍스트 데이터
Token type	fernet, uuid, pki, pkiz
	어떤 자원에 접근이 가능한지 범위가 지정되어 있음 (시간 제한 있음)
Project	Keystone V2까지 Tenant라는 이름으로 사용 (V3 이후 Project)
	어떤 자원이나 어플리케이션에 대한 권리를 가진 보안그룹
	프로젝트는 특정 도메인에 의해 소유
Endpoint	사용자가 서비스를 이용하기 위해 연결정보를 제공하는 접근 가능한 네트워크 주소 (URL)
EndPoint type	admin, internal, public
Role	사용자가 어떤 동작을 수행하도록 허용하는 규칙
	사용자가 가지는 역할은 사용자에게 발행된 토큰을 통해 확인
	사용자가 서비스를 호출하면, 서비스는 토큰에 저장된 사용자의 역할을 해석하여 허용유무 결정
Domain	구성요소를 효과적으로 관리하기 위한 사용자, 그룹, 프로젝트의 집합
	사용자들은 한 도메인의 관리자의 권한 등을 부여받는 방식으로 역할을 부여가능

- 각 구성요소의 개념과 관계
- Keystone은 위에서 언급하였듯이 사용자 인증 부분과 서비스 인증 부분을 관리
- 사용자일 때는 사용자 ID와 패스워드, 사용자 권한의 롤(Roll)을 등록
- 서비스일 때는 서비스를 등록하고 해당 서비스의 엔드포인트 URL을 등록
 - 도메인(Domain)은 서로 분리되어 있음
 - 각 도메인에는 프로젝트와 사용자가 존재
 - 프로젝트는 사용자를 가질 수 있음
 - 사용자에게는 롤이 있으며, 여러 프로젝트의 구성원이 될 수 있음

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

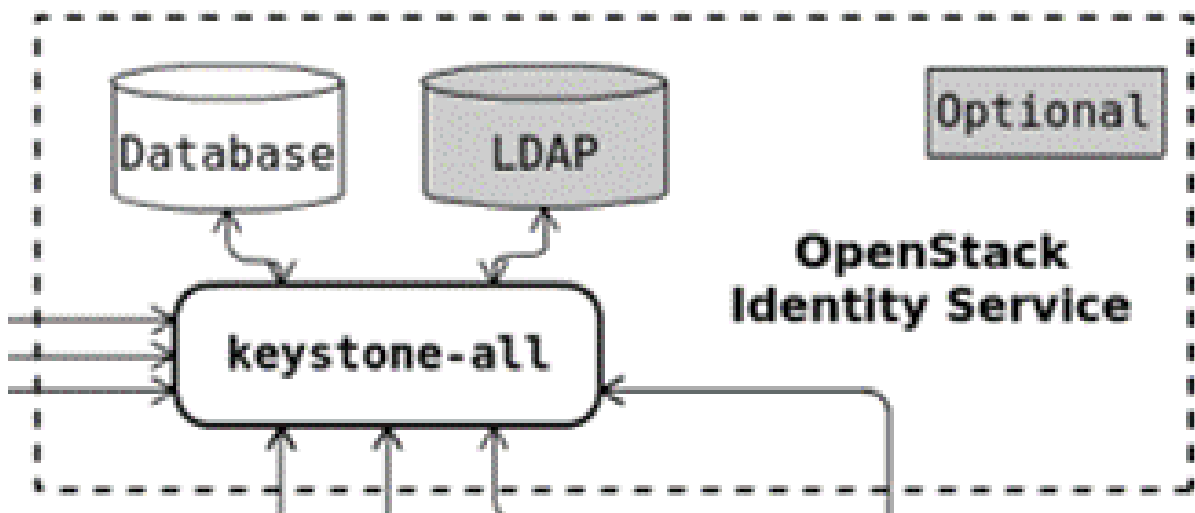
- 관리자 롤(Admin Role)을 가진 사용자끼리, 일반 사용자 롤(Member Role)을 가진 사용자간의 그룹핑(Grouping)이 가능

Keystone의 논리 아키텍처


- Keystone의 논리 아키텍처는 Token, Catalog, Policy, Identity로 구성되어져있다.

구성요소	역할
Token Backend	사용자별 토큰을 관리
Catalog Backend	오픈스택에서 모든 서비스의 엔드포인트 URL을 관리
Policy Backend	테넌트, 사용자 계정, 롤 등을 관리
Identity Backend	사용자 인증을 관리

Keystone의 OpenStack 에서의 역할



- Keystone은 OpenStack에서 모든 서비스를 관장하는 위치에서 역할을 수행
- 모든 User, Service는 Keystone의 인증을 통해서만 요청이 가능하며, 응답이 가능
- Keystone은 타인이나 해커에게서 시스템을 안전하게 보호하고 사용자 등록, 삭제, 권한 관리, 사용자가 접근할 수 있는 서비스 포인트 관리와 다른 API 인증 등의 전체적인 인증

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

프로세스를 관리하는 역할을 수행

Keystone Install

- Keystone User, DB 생성

```
$ controller> mysql -u root -p
$ MariaDB> create database keystone;
$ MariaDB> grant all privileges on keystone.* to
keystone@'localhost' identified by 'qwer1234';
$ MariaDB> grant all privileges on keystone.* to keystone@'%'
identified by 'qwer1234';
$ MariaDB> flush privileges;
$ MariaDB> exit;
```

- Keystone install

```
$ controller> dnf --enablerepo=centos-openstack-
ussuri,epel,PowerTools -y install openstack-keystone python3-
openstackclient httpd mod_ssl python3-mod_wsgi python3-oauth2client
# keystone 및 관련 모듈을 설치


$ controller> vi /etc/keystone/keystone.conf
[cache]
memcache_servers = controller:11211

[database]
connection = mysql+pymysql://keystone:qwer1234@controller/keystone

[token]
provider = fernet

$ controller> su -s /bin/bash keystone -c "keystone-manage db_sync"
$ controller> keystone-manage fernet_setup --keystone-user keystone
--keystone-group keystone
$ controller> keystone-manage credential_setup --keystone-user
keystone --keystone-group keystone
# Keystone DB 임포트

$ controller> keystone-manage bootstrap --bootstrap-password
qwer1234 \
--bootstrap-admin-url http://controller:5000/v3/ \
--bootstrap-internal-url http://controller:5000/v3/ \
--bootstrap-public-url http://controller:5000/v3/ \
--bootstrap-region-id RegionOne
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ controller> setsebool -P httpd_use_openstack on
$ controller> setsebool -P httpd_can_network_connect on
$ controller> setsebool -P httpd_can_network_connect_db on
$ controller> vi keystone-httpd.te
module keystone-httpd 1.0;

require {
    type httpd_t;
    type keystone_log_t;
    class file create;
    class dir { add_name write };
}

#===== httpd_t =====
allow httpd_t keystone_log_t:dir { add_name write };
allow httpd_t keystone_log_t:file create;


$ controller> checkmodule -m -M -o keystone-httpd.mod keystone-
httpd.te
$ controller> semodule_package --outfile keystone-httpd.pp --module
keystone-httpd.mod
$ controller> semodule -i keystone-httpd.pp
$ controller> firewall-cmd --add-port=5000/tcp --permanent
$ controller> firewall-cmd --reload
# 방화벽 및 SELinux를 설정

$ controller> vi /etc/httpd/conf/httpd.conf
ServerName controller:80
# 99 번 줄에 추가

$ controller> ln -s /usr/share/keystone/wsgi-keystone.conf
/etc/httpd/conf.d/
$ controller> systemctl enable --now httpd
# httpd 서비스를 등록
```

- Keystone Project 생성

```
$ controller> vi ~/admin_key
export OS_PROJECT_DOMAIN_NAME=default
export OS_USER_DOMAIN_NAME=default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=qwer1234
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
export PS1='[\u@\h \W~(keystone)]\$ '
$ controller> chmod 600 ~/admin_key
$ controller> source ~/admin_key
$ controller> echo "source ~/admin_key " >> ~/.bash_profile
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


```
# keystone 인증파일 생성 후 시작시 등록되게 등록시킵니다.

$ controller ~(keystone)> openstack project create --domain default
--description "Service Project" service
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| description | Service Project                         |
| domain_id   | default                                 |
| enabled     | True                                    |
| id          | 7c10c02365be496fb47f12bfd40fe4a7      |
| is_domain   | False                                  |
| name        | service                                |
| options     | {}                                     |
| parent_id   | default                                 |
| tags        | []                                     |
+-----+-----+

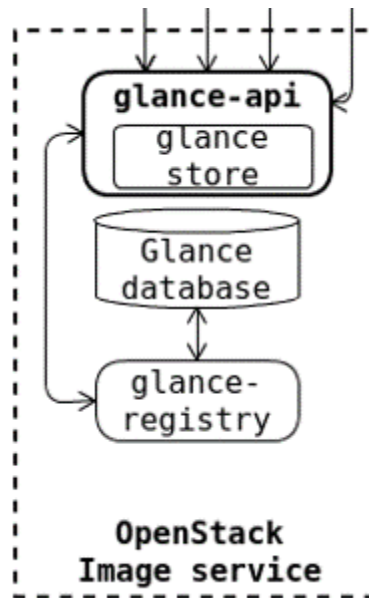
$ controller ~(keystone)> openstack project list
+-----+-----+
| ID                  | Name    |
+-----+-----+
| 7c10c02365be496fb47f12bfd40fe4a7 | service |
| c76211c24a1f460ca67274d655d46725 | admin   |
+-----+-----+
```

이미지를 관리하는 서비스 : Glance

- Cloud Computing을 사용하기 위해서는 Virtual Machine을 생성하기 위한 이미지가 필요로 하며, Glance는 Nova에서 생성하는 인스턴스의 운영체제에 해당하는 이미지를 관리하는 역할을 수행하는 서비스이다.

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


Glance의 구성요소



구성요소	역할
Glance-api	이미지를 확인/ 복구/ 저장하는 등의 질의를 하기 위한 api 요청/ 응답을 담당
Glance-registry	이미지에 대한 메타데이터를 저장하고 처리하는 역할을 담당 및 Glance database에 저장된 데이터를 불러들이는 역할을 수행
Glance-database	이미지의 관련 정보들을 보관

Glance의 논리 아키텍처

- Glance 사용자들은 Glance-api로 이미지를 등록, 삭제 관리
- Glance-api는 Glance-registry와 Glance database에서 이미지를 관리
- 이미지를 등록할 때는 glance-registry로 Glance database에 저장
- 등록된 이미지를 사용할 때는 Glance database에 바로 사용을 요청
- 관리자는 운영하려는 운영체제의 이미지를 Glance-registry로 Glance database에 등록

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


Glance의 Image

- 가상 머신 이미지 포맷
 - aki: 아마존 커널 이미지
 - ami: 아마존 머신 이미지
 - ari: 아마존 ram 디스크 이미지
 - iso: 광학 디스크나 CD-ROM 의 데이터 콘텐츠를 지원하는 아카이브 포맷
 - qcow2: QEMU 에뮬레이터가 지원하는 포맷, 동적으로 확장할 수 있으며, Copy on Write 를 지원
 - raw: 구조화되지 않은 디스크 포맷
 - vdi: VirtualBox 모니터와 QEMU 에뮬레이터가 지원하는 디스크 포맷
 - vhd: VHD 디스크 포맷은 VMware, Xen 마이크로소프트, VirtualBox 같은 가상 머신 모니터가 사용하는 일반적인 디스크 포맷
 - vhdx: VHDX 디스크 포맷은 큰 디스크 크기를 지원하는 VHD 형식의 향상된 버전
 - vmdk: 일반적인 디스크 포맷으로 여러 가상 머신 모니터가 지원

- 컨테이너 포맷(container Format)
 - aki: 아마존 커널 이미지
 - ami: 아마존 머신 이미지
 - bare: 아마존 ram 디스크 이미지
 - docker: Docker 컨테이너 포맷
 - ova: tar 파일의 OVF 패키지
 - ovf: OVF 컨테이너 포맷

Glance Install

- Glance Service, User, DB 생성


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ contoller ~(keystone)> openstack user create --domain default --
project service --password qwer1234 glance
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| default_project_id | 7c10c02365be496fb47f12bfd40fe4a7 |
| domain_id       | default                               |
| enabled         | True                                 |
| id              | 03f5b16a7be84cb688617d1943c8fe8c |
| name            | glance                              |
| options         | {}                                  |
| password_expires_at | None                               |
+-----+-----+

$ contoller ~(keystone)> openstack role add --project service --user
glance admin
$ contoller ~(keystone)> openstack service create --name glance --
description "OpenStack Image service" image
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| description    | OpenStack Image service               |
| enabled        | True                                 |
| id             | af365771c17a4a25ae1d0c659e2dc0eb |
| name           | glance                              |
| type           | image                               |
+-----+-----+

$ contoller ~(keystone)> openstack endpoint create --region
RegionOne image public http://controller:9292
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| enabled        | True                                 |
| id             | cc65faecd7b042ffafd0f262cd7547df |
| interface      | public                             |
| region         | RegionOne                          |
| region_id      | RegionOne                          |
| service_id     | af365771c17a4a25ae1d0c659e2dc0eb |
| service_name   | glance                             |
| service_type   | image                              |
| url            | http://controller:9292             |
+-----+-----+

$ contoller ~(keystone)> openstack endpoint create --region
RegionOne image internal http://controller:9292
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| enabled        | True                                 |
| id             | ea41c7b17c844e658ac83c547eddcf6d |
| interface      | internal                           |
| region         | RegionOne                          |
| region_id      | RegionOne                          |
+-----+-----+
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| service_id | af365771c17a4a25ae1d0c659e2dc0eb |
| service_name | glance |
| service_type | image |
| url | http://controller:9292 |
+-----+-----+
$ controller ~(keystone)> openstack endpoint create --region
RegionOne image admin http://controller:9292
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | 1393a64ef0ec428ba437602ac5b390f6 |
| interface | admin |
| region | RegionOne |
| region_id | RegionOne |
| service_id | af365771c17a4a25ae1d0c659e2dc0eb |
| service_name | glance |
| service_type | image |
| url | http://controller:9292 |
+-----+-----+

$ controller> mysql -u root -p
$ MariaDB> create database glance;
$ MariaDB> grant all privileges on glance.* to glance@'localhost'
identified by 'qwer1234';
$ MariaDB> grant all privileges on glance.* to glance@ '%' identified
by 'qwer1234';
$ MariaDB> flush privileges;
$ MariaDB> exit;

```

- Glance Install

```

$ controller> dnf --enablerepo=centos-openstack-
ussuri,PowerTools,epel -y install openstack-glance
# Glance 및 관련 모듈을 설치


$ controller> vi /etc/glance/glance-api.conf
[DEFAULT]
bind_host = 0.0.0.0

[glance_store]
stores = file,http
default_store = file
filesystem_store_datadir = /var/lib/glance/images/

[database]
connection = mysql+pymysql://glance:qwer1234@controller/glance

[keystone_auth_token]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = glance
password = qwer1234

[paste_deploy]
flavor = keystone
# glance 파일을 수정

$ controller> su -s /bin/bash glance -c "glance-manage db_sync"
$ controller> systemctl enable --now openstack-glance-api
# Glance DB를 импорт 시킨후 서비스를 등록

$ controller> setsebool -P glance_api_can_network on
$ controller> vi glanceapi.te
module glanceapi 1.0;

require {
    type glance_api_t;
    type httpd_config_t;
    type iscsid_exec_t;
    class dir search;
    class file { getattr open read };
}

#===== glance_api_t =====
allow glance_api_t httpd_config_t:dir search;
allow glance_api_t iscsid_exec_t:file { getattr open read };

$ controller> checkmodule -m -M -o glanceapi.mod glanceapi.te
$ controller> semodule_package --outfile glanceapi.pp --module glanceapi.mod
$ controller> semodule -i glanceapi.pp
$ controller> firewall-cmd --add-port=9292/tcp --permanent
$ controller> firewall-cmd --reload

```

- Image upload

```

$ controller ~(keystone)> mkdir -p /var/kvm/images
$ controller ~(keystone)> wget http://download.cirros-
cloud.net/0.5.1/cirros-0.5.1-x86_64-disk.img
# cirros 이미지를 다운로드


```

- image import

```

$ controller ~(keystone)> openstack image create "cirros" --file cirros-0.5.1-x86_64-disk.img --disk-format qcow2

```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

+-----+-----+
| Field           | Value                                     |
+-----+-----+
| checksum        | 1d3062cd89af34e419f7100277f38b2b      |
| container_format | bare                                    |
| created_at      | 2020-08-06T11:08:39Z                  |
| disk_format     | qcow2                                  |
| file            | /v2/images/dc7c2474-8ec9-4f74-a1c3-7cf6a9ad3d16/file |
| id              | dc7c2474-8ec9-4f74-a1c3-7cf6a9ad3d16 |
| min_disk        | 0                                       |
| min_ram         | 0                                       |
| name            | Cirros                                 |
| owner           | c76211c24a1f460ca67274d655d46725     |
| properties      | os_hash_algo='sha512', ...           |
| protected       | False                                  |
| schema          | /v2/schemas/image                    |
| size            | 16338944                               |
| status          | active                                 |
| tags            |                                         |
| updated_at      | 2020-08-06T11:08:39Z                  |
| visibility      | shared                                 |
+-----+-----+
# 이미지를 등록

$ controller ~(keystone)> openstack image list
+-----+-----+-----+
| ID              | Name    | Status |
+-----+-----+-----+
| dc7c2474-8ec9-4f74-a1c3-7cf6a9ad3d16 | Cirros  | active |
+-----+-----+-----+
# 이미지를 확인합니다.

```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

가상의 서버를 생성하는 서비스 : Nova

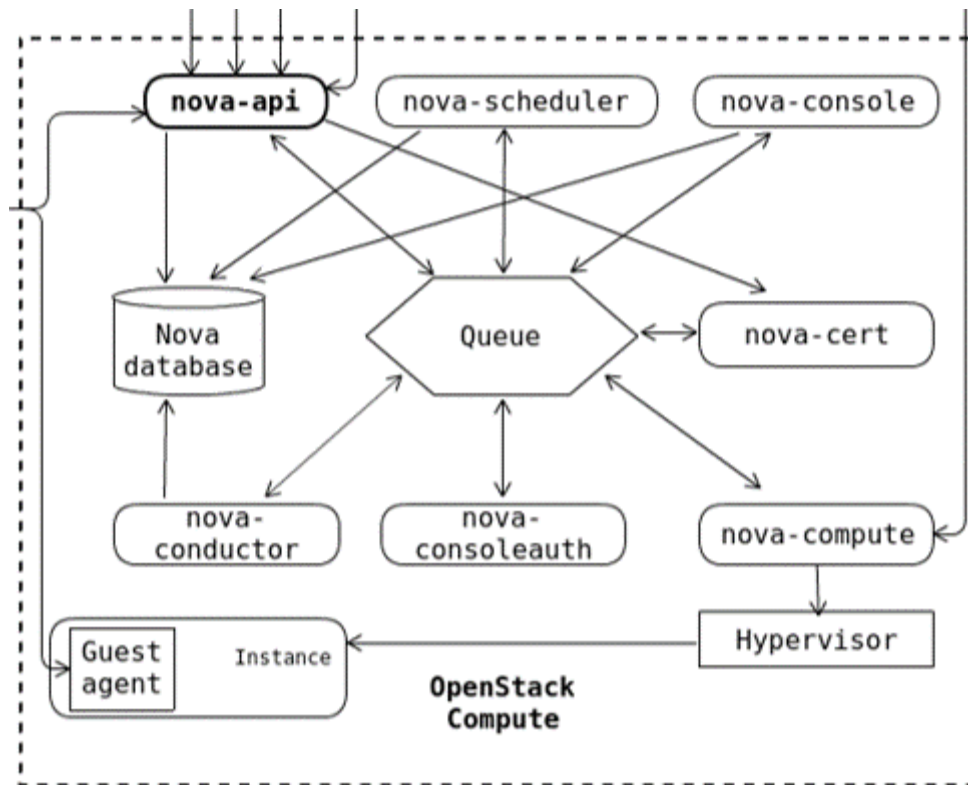
- Nova는 Compute 서비스의 핵심
- Compute 서비스란, 가상머신이 필요한 자원을 할당하고, 관리하는 서비스로 하이퍼바이저, 메시지 Queue, 인스턴스 접속을 하는 콘솔 등의 다양한 기능이 유기적으로 연결되어 가상 서버를 생성할 수 있는 시스템을 구성하는 서비스

Nova의 고려요소


고려사항	설명
CPU	compute 서비스가 동작할 호스트 시스템의 cpu가 기본적으로 자체 하드웨어 가상화를 지원이 필수
Hypervisor	서비스에 사용할 하이퍼바이저를 맞게 설정해야 하며, 기본적으로 사용하는 Hypervisor은 KVM/QEMU
Storage	compute 서비스를 통해 인스턴스가 생성되면서 시스템의 디스크 용량의 제한을 가할 수 있음, 이를 위해 넉넉한 공간이 필요
Overcommit	기본적으로 자원을 할당하는 경우 1:1이 아닌 CPU는 16:1, Memory는 1.5:1로 할당 되어짐
네트워킹	생성된 인스턴스의 경우 nova가 독자적으로 구현하는 것이 아닌 다른 network 서비스를 연계해서 사용해야하며, 주로 Neutron 네트워크 서비스와 함께 사용

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Nova의 논리 아키텍처 및 구성요소



서비스	역할
nova-api	최종 사용자가 API콜을 통해 서비스 간 질의 응답을 담당
nova-compute	가상화 API를 이용하여 가상 머신 인스턴스를 생성하고 종료하는 역할을 수행
nova-scheduler	compute host가 다수인 경우 큐를 통해 받은 메시지를 누구에게 명령할 것인지를 결정
nova-conductor	코디네이션과 데이터베이스 쿼리를 지원하는 서버 데몬
nova-cert	X509 인증서에 대한 Nova Cert서비스를 제공하는 서버 데몬
nova-consoleauth	데몬, 콘솔 프록시를 제공하는 사용자에게 인증 토큰 제공
Guest Agent	실제 compute 시스템 상에 구축된 인스턴스로 Nova-compute 서비스에 의해 제어되어짐
nova-api-metadata	인스턴스의 메타데이터의 요청을 처리
nova-novncproxy	VNC 콘솔화면을 제공
nova-novaclient	nova REST API를 사용하는 클라이언트 프로그램
nova-network	인스턴스의 네트워크 기능을 수행

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

nova-compute-kvm	인스턴스(가상 머신)와 관련된 모든 프로세스를 처리
python-guestfs	파일 생성 기능을 지원하는 Python 라이브러리
qemu-kvm	KVM 하이퍼바이저

- Nova는 위의 표와 같이 다수의 서비스들이 존재
- Nova는 대시보드나 콘솔에서 호출하는 nova-api에서 시작되며 Queue를 이용해 nova-compute에 인스턴스를 생성하라는 명령을 전달
- nova-compute는 하이퍼바이저 라이브러리를 이용해 하이퍼바이저에 인스턴스를 생성하려는 명령어를 전달
- Hypervisor을 통해 인스턴스를 생성
- 생성된 인스턴스는 nova-api로 접근할 수 있으며, Nova의 모든 기능은 메시지 Queue로 처리가능


Nova 하이퍼바이저의 종류

- 기본 하이퍼바이저는 KVM과 QEMU를 지원
- 프로바이더가 테스트하는 Hyper-V, VMware, XenServer, Sen via libvirt
- 몇 번의 테스트만 하는 하이퍼바이저 드라이버인 베어메탈, Docker, LXC via libvirt가 존재

Nova Install

- Nova 및 Ceilometer Service User, Service, DB 생성

```
$ controller ~(keystone)> openstack user create --domain default --project service --password qwer1234 nova
+-----+-----+-----+-----+
| Field          | Value                                     |
+-----+-----+-----+-----+
| default_project_id | 7c10c02365be496fb47f12bfd40fe4a7 |
| domain_id        | default                                 |
| enabled          | True                                   |
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| id | f26027517d5e4b5b984b5db8d42398c8 |
| name | nova |
| options | {} |
| qwer1234_expires_at | None |
+-----+-----+

$ controller ~(keystone)> openstack user create --domain default --
project service --password qwer1234 placement
+-----+-----+
| Field | Value |
+-----+-----+
| default_project_id | 7c10c02365be496fb47f12bfd40fe4a7 |
| domain_id | default |
| enabled | True |
| id | 2394500b4512456f9d9d5066a5ecb1f7 |
| name | placement |
| options | {} |
| qwer1234_expires_at | None |
+-----+-----+


$ controller ~(keystone)> openstack role add --project service --
user nova admin
$ controller ~(keystone)> openstack role add --project service --
user placement admin

$ controller ~(keystone)> openstack service create --name nova --
description "OpenStack Compute service" compute
+-----+-----+
| Field | Value |
+-----+-----+
| description | OpenStack Compute service |
| enabled | True |
| id | 28d495eca718439f9dc6ce395e0720dc |
| name | nova |
| type | compute |
+-----+-----+

$ controller ~(keystone)> openstack service create --name placement
--description "OpenStack Compute Placement service" placement
+-----+-----+
| Field | Value |
+-----+-----+
| description | OpenStack Compute Placement service |
| enabled | True |
| id | 8515d3d046834de9b71b2938aae89898 |
| name | placement |
| type | placement |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne compute public http://controller:8774/v2.1/%(tenant_id)s
+-----+-----+
| Field | Value |
+-----+-----+

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```


| enabled      | True |
| id           | f13ca97a20eb46a3a1c1dfab546a00cc |
| interface    | public |
| region       | RegionOne |
| region_id    | RegionOne |
| service_id   | 28d495eca718439f9dc6ce395e0720dc |
| service_name | nova |
| service_type | compute |
| url          | http://controller:8774/v2.1/$(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne compute internal
http://controller:8774/v2.1/$(tenant_id)s
+-----+-----+
| Field      | Value |
+-----+-----+
| enabled    | True |
| id         | 1bc41c829f2f47e7962cba46f0da8ddc |
| interface  | internal |
| region     | RegionOne |
| region_id  | RegionOne |
| service_id | 28d495eca718439f9dc6ce395e0720dc |
| service_name | nova |
| service_type | compute |
| url        | http://controller:8774/v2.1/$(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne compute admin http://controller:8774/v2.1/$(tenant_id)s
+-----+-----+
| Field      | Value |
+-----+-----+
| enabled    | True |
| id         | 8022a415f22c400c92989320a2be3133 |
| interface  | admin |
| region     | RegionOne |
| region_id  | RegionOne |
| service_id | 28d495eca718439f9dc6ce395e0720dc |
| service_name | nova |
| service_type | compute |
| url        | http://controller:8774/v2.1/$(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne placement public http://controller:8778
+-----+-----+
| Field      | Value |
+-----+-----+
| enabled    | True |
| id         | 5e988f2be72242f0b3923e27e9db009c |
| interface  | public |
| region     | RegionOne |
| region_id  | RegionOne |


```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| service_id | 8515d3d046834de9b71b2938aae89898 |
| service_name | placement |
| service_type | placement |
| url | http://controller:8778 |
+-----+-----+
$ controller ~(keystone)> openstack endpoint create --region
RegionOne placement internal http://controller:8778
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | a68cf8b6eeb043c2aa1ec95d7711cb50 |
| interface | internal |
| region | RegionOne |
| region_id | RegionOne |
| service_id | 8515d3d046834de9b71b2938aae89898 |
| service_name | placement |
| service_type | placement |
| url | http://controller:8778 |
+-----+-----+
$ controller ~(keystone)> openstack endpoint create --region
RegionOne placement admin http://controller:8778
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | 63e47fcbfd7841dd95bb4d9d9a910ab5 |
| interface | admin |
| region | RegionOne |
| region_id | RegionOne |
| service_id | 8515d3d046834de9b71b2938aae89898 |
| service_name | placement |
| service_type | placement |
| url | http://controller:8778 |
+-----+-----+
$ controller> mysql -u root -p
$ MariaDB> create database nova;
$ MariaDB> grant all privileges on nova.* to nova@'localhost'
identified by 'qwer1234';
$ MariaDB> grant all privileges on nova.* to nova@'%' identified by
'qwer1234';
$ MariaDB> create database nova_api;
$ MariaDB> grant all privileges on nova_api.* to nova@'localhost'
identified by 'qwer1234';
$ MariaDB> grant all privileges on nova_api.* to nova@'%' identified
by 'qwer1234';
$ MariaDB> create database nova_cell0;
$ MariaDB> grant all privileges on nova_cell0.* to nova@'localhost'
identified by 'qwer1234';

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ MariaDB> grant all privileges on nova_cell0.* to nova@'%'
identified by 'qwer1234';
$ MariaDB> create database placement;
$ MariaDB> grant all privileges on placement.* to
placement@'localhost' identified by 'qwer1234';
$ MariaDB> grant all privileges on placement.* to placement@'%'
identified by 'qwer1234';
$ MariaDB> flush privileges;
$ MariaDB> exit;
```

- Nova Install (controller)

```
$ controller> dnf --enablerepo=centos-openstack-
ussuri,PowerTools,epel -y install openstack-nova openstack-
placement-api
# nova 및 관련 모듈을 설치

$ controller> vi /etc/nova/nova.conf
[DEFAULT]
my_ip = 192.168.10.10
# my_ip = controller IP

state_path = /var/lib/nova
enabled_apis = osapi_compute,metadata
log_dir = /var/log/nova
transport_url = rabbit://openstack:qwer1234@controller

[api]
auth_strategy = keystone

[glance]
api_servers = http://controller:9292


[vnc]
enabled = true
server_listen = $my_ip
server_proxyclient_address = $my_ip

[oslo_concurrency]
lock_path = $state_path/tmp

[api_database]
connection = mysql+pymysql://nova:qwer1234@controller/nova_api

[database]
connection = mysql+pymysql://nova:qwer1234@controller/nova

[keystone_auth_token]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = nova
password = qwer1234

[placement]
auth_url = http://controller:5000
os_region_name = RegionOne
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = placement
password = qwer1234

[wsgi]
api_paste_config = /etc/nova/api-paste.ini

$ controller> vi /etc/placement/placement.conf
[DEFAULT]
debug = false

[api]
auth_strategy = keystone


[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = placement
password = qwer1234

[placement_database]
connection = mysql+pymysql://placement:qwer1234@controller/placement

$ controller> vi /etc/httpd/conf.d/00-placement-api.conf
<Directory /usr/bin>
    Require all granted
</Directory>
# 15 번 줄에 추가

$ controller> su -s /bin/bash placement -c "placement-manage db
sync"

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ controller> su -s /bin/bash nova -c "nova-manage api_db sync"
$ controller> su -s /bin/bash nova -c "nova-manage cell_v2
map_cell0"
$ controller> su -s /bin/bash nova -c "nova-manage db sync"
$ controller> su -s /bin/bash nova -c "nova-manage cell_v2
create_cell --name cell1"
# nova DB import

$ controller> semanage port -a -t http_port_t -p tcp 8778
$ controller> firewall-cmd --add-
port={6080/tcp,6081/tcp,6082/tcp,8774/tcp,8775/tcp,8778/tcp} --
permanent
$ controller> firewall-cmd --reload
$ controller> systemctl restart httpd
$ controller> chown placement. /var/log/placement/placement-api.log
$ controller> for service in api conductor scheduler novncproxy; do
systemctl enable --now openstack-nova-$service
done
# Selinux 및 방화벽을 설정

$ controller ~(keystone)> openstack compute service list
+-----+-----+-----+-----+-----+-----+
| ID | Binary           | Host           | Zone   | Status | State |
| Updated At |
+-----+-----+-----+-----+-----+-----+
| 4 | nova-conductor   | controller     | internal | enabled | up     |
| 2020-08-06T12:10:34.000000 |
| 5 | nova-scheduler   | controller     | internal | enabled | up     |
| 2020-08-06T12:10:38.000000 |
+-----+-----+-----+-----+-----+-----+
```

- Nova KVM Install (Compute)


```
$ compute> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install qemu-kvm libvirt virt-install libguestfs-tools
# KVM 관련 모듈을 설치

$ compute> systemctl enable --now libvirtd
# libvirtd 서비스를 등록 및 시작

$ compute> nmcli connection add type bridge autoconnect yes con-name
br0 ifname br0
# br0 의 가상 브릿지를 추가합니다.

$ compute> nmcli connection modify br0 ipv4.addresses 10.10.10.30/24
ipv4.method manual
# 가상 브릿지의 IP 를 추가 ( compute node ip )

$ compute> nmcli connection modify br0 ipv4.gateway 10.10.10.10
# 가상 브릿지의 GATEWAY 를 등록
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ compute> nmcli connection modify br0 ipv4.dns 8.8.8.8
# 가상 브릿지의 DNS 를 등록

$ compute> nmcli connection del ens34
# 본래의 네트워크 인터페이스를 삭제

$ compute> nmcli connection add type bridge-slave autoconnect yes
con-name ens160 ifname ens160 master br0
# 삭제한 네트워크 인터페이스 대신 브릿지를 매핑시키고 네트워크를 재시작
```

- Nova Install (Compute)

```
$ compute> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install openstack-nova-compute
# nova 및 관련 모듈을 설치

$ controller> scp /etc/nova/nova.conf compute:/etc/nova/nova.conf
# nova 의 기본설정파일을 복사

$ compute> vi /etc/nova/nova.conf
[default]
my_ip = 192.168.10.30
# 현재의 my_ip 는 compute 의 IP

[libvirt]
virt_type = qemu


[vnc]
enabled = True
server_listen = 0.0.0.0
server_proxyclient_address = $my_ip
novncproxy_base_url = http://controller:6080/vnc_auto.html
# nova 관련 설정을 추가합니다.

$ compute> firewall-cmd --add-port=5900-5999/tcp --permanent
$ compute> firewall-cmd --reload
$ compute> systemctl enable --now libvirtd
$ compute> systemctl enable --now openstack-nova-compute
```

- Nova 설치확인

```
$ controller> su -s /bin/bash nova -c "nova-manage cell_v2
discover_hosts"
# DB 에 compute 의 대한 설정을 업데이트


$ controller> nova-manage cell_v2 discover_hosts --verbose
# 만약 compute 노드가 검색이 안되었을 시 추가적으로 검색
```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

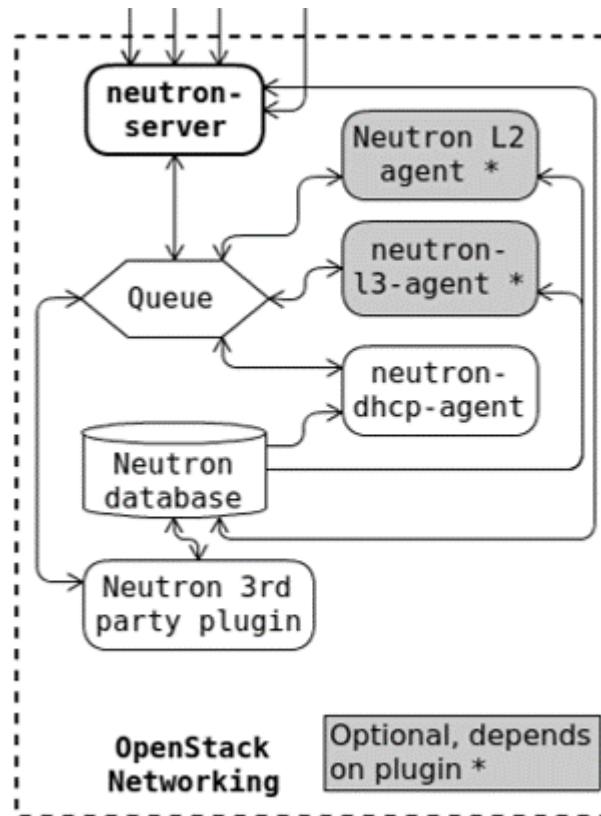
```
$ controller ~(keystone)> openstack compute service list
+-----+-----+-----+-----+-----+-----+
| ID | Binary          | Host          | Zone     | Status  | State |
Updated At
+-----+-----+-----+-----+-----+-----+
| 4 | nova-conductor  | controller    | internal | enabled | up    |
2020-08-06T21:40:34.000000 |
| 5 | nova-scheduler  | controller    | internal | enabled | up    |
2020-08-06T21:40:37.000000 |
| 8 | nova-compute     | compute       | nova     | enabled | up    |
2020-08-06T21:40:36.000000 |
+-----+-----+-----+-----+-----+-----+
```

네트워크를 관리하는 서비스 : Neutron


- Neutron은 네트워크 서비스로 여러 노드에 다양한 프로세스를 배치하는 독립형 서비스를 의미한다.
- Neutron은 서로 및 다른 OpenStack의 서비스와 상호작용이 가능하며 ovs, linux-bridge 등의 SDN 기술을 사용한다.

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Neutron의 논리 아키텍처 및 구성요소

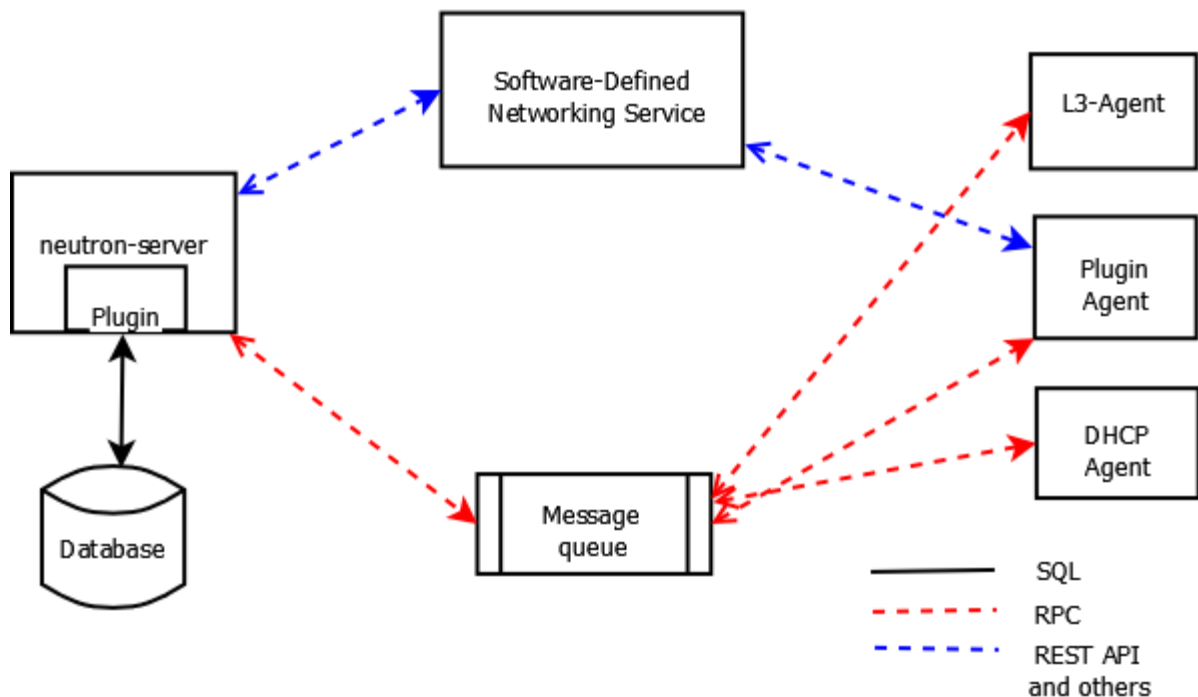


구성요소	기능
neutron-server	network api의 기능 및 네트워크 확장 기능을 서비스하며, 각 포트의 대한 모델 및 Pfmf 지정, AMQP를 사용하여 데이터베이스와 통신하는 플러그인을 통해 수행
neutron-L2-agent	OVS 가상 Bridge 사이에서데이터 패킷을 전달하기 위한 중계장치
neutron-l3-agent	테넌트 네트워크에서 VM의 외부 네트워크 엑세서를 위한 L3/ NAT 전달을 제공
neutron-dhcp-agent	테넌트 네트워크에 DHCP 서비스를 제공, DHCP agent는 메시지 큐에 엑세스할 수 있는 권한이 필요
Queue	다른 서비스 간의 통신의 역할을 수행
Neutron Database	Neutron 서비스를 수행하기 위한 일련의 정보들은 보관, 관리하는 DB
Neutron 3rd Party Plugin	Neutron 서비스의 안정적인 통신 역할을 수행
plugin agent	각 compute node에서 실행되며 로컬 vswitch을 구성 및 관리


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- Neutron은 다양한 네트워크 플러그인이나 네트워크 모델을 지원
- 사용자는 Neutron API를 이용해 neutron-server로 IP할당을 요청 후 Neutron-server에 들어온 요청을 Queue로 다시 리다이렉트를 진행
- Queue는 Neutron-dhcp-agent와 Neutron 3rd Party plugin으로 IP 할당 지시를 내리는 역할을 수행
- Neutron-dhcp-agent와 Neutron 3rd Party Plugin은 지시받은 작업을 수행
- Neutron-server는 수시로 작업 상태를 Neutron databases에 저장
- 할당된 IP를 인스턴스에서 사용 가능

Neutron의 네트워킹 프로세스

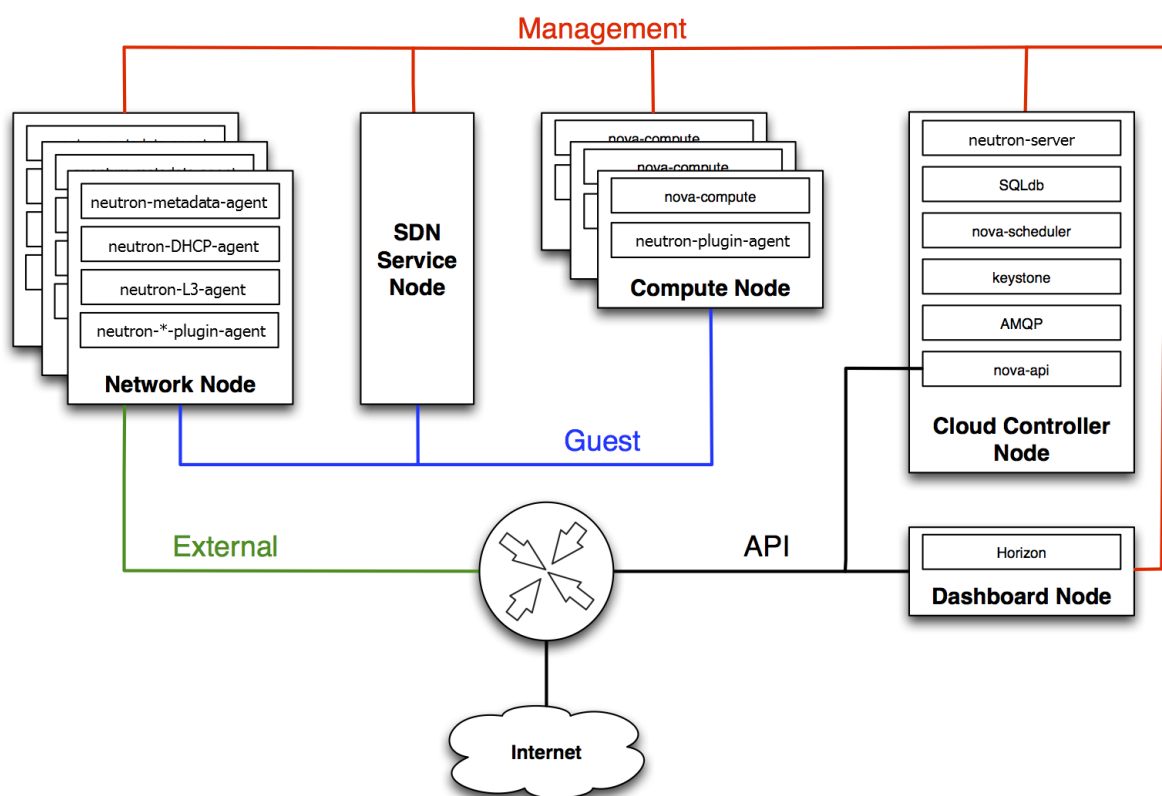


- Neutron-server에 의해 명령을 요청 받음


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- Plugin을 토대로 Message queue를 통해 각 agent에 기능을 수행
- 이와 함께 SDN 서비스를 수행하여 다중서비스 제공

Neutron network의 종류



네트워크의 종류	기능
Management network	OpenStack 구성 요소 간의 내부 통신에 사용, 기본적으로 IP 주소는 데이터 센터 내에서만 사용이 가능
Guest network	클라우드 배포 내에서 인스턴스 데이터 통신에 사용되며, 네트워킹 플러그인 및 테넌트가 만든 가상 네트워크의 구성 선택에 따라 변동
External network	외부에서 인스턴스에 대한 액세스를 위해 제공되는 네트워크
API network	OpenStack API를 외부에 노출시키는 네트워크

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| description | OpenStack Networking service |
| enabled    | True                          |
| id         | 055e5f6e38004338b0ae4a86e77932ae |
| name       | neutron                      |
| type       | network                      |
+-----+-----+


# neutron service 및 user 을 생성합니다.

$ controller ~(keystone)> openstack endpoint create --region
RegionOne network public http://controller:9696
+-----+-----+
| Field      | Value                          |
+-----+-----+
| enabled    | True                          |
| id         | 350c666f597a41e59234b09f534aa72f |
| interface  | public                        |
| region     | RegionOne                     |
| region_id  | RegionOne                     |
| service_id | 055e5f6e38004338b0ae4a86e77932ae |
| service_name | neutron                      |
| service_type | network                      |
| url        | http://controller:9696       |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne network internal http://controller:9696
+-----+-----+
| Field      | Value                          |
+-----+-----+
| enabled    | True                          |
| id         | b9cad959e1634ff797e27f00d50e9578 |
| interface  | internal                      |
| region     | RegionOne                     |
| region_id  | RegionOne                     |
| service_id | 055e5f6e38004338b0ae4a86e77932ae |
| service_name | neutron                      |
| service_type | network                      |
| url        | http://controller:9696       |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne network admin http://controller:9696
+-----+-----+
| Field      | Value                          |
+-----+-----+
| enabled    | True                          |
| id         | 72fc145deb1d4d508e3691b3bf77708e |
| interface  | admin                        |
| region     | RegionOne                     |
| region_id  | RegionOne                     |
| service_id | 055e5f6e38004338b0ae4a86e77932ae |

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| service_name | neutron |
| service_type | network |
| url          | http://controller:9696 |
+-----+-----+
```

```
$ controller> mysql -u root -p
$ MariaDB> create database neutron_ml2;
$ MariaDB> grant all privileges on neutron_ml2.* to
neutron@'localhost' identified by 'qwer1234';
$ MariaDB> grant all privileges on neutron_ml2.* to neutron@'%'
identified by 'qwer1234';
$ MariaDB> flush privileges;
$ MariaDB> exit;
```

- Neutron Install (controller)


```
$ controller> dnf --enablerepo=centos-openstack-
ussuri,PowerTools,epel -y install openstack-neutron openstack-
neutron-ml2
# neutron 및 관련 모듈을 설치

$ controller> vi /etc/neutron/neutron.conf
[DEFAULT]
core_plugin = ml2
service_plugins = router
auth_strategy = keystone
state_path = /var/lib/neutron
dhcp_agent_notification = True
allow_overlapping_ips = True
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
transport_url = rabbit://openstack:qwer1234@controller

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = neutron
password = qwer1234

[database]
connection = mysql+pymysql://neutron:qwer1234@controller/neutron_ml2

[nova]
auth_url = http://controller:5000
auth_type = password
project_domain_name = default
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

user_domain_name = default
region_name = RegionOne
project_name = service
username = nova
password = qwer1234

[oslo_concurrency]
lock_path = $state_path/tmp

$ controller> vi /etc/neutron/metadata_agent.ini
[DEFAULT]
nova_metadata_host = controller
metadata_proxy_shared_secret = metadata_secret

[cache]
memcache_servers = controller:11211

$ controller> vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,vlan,gre,vxlan
tenant_network_types = vxlan
mechanism_drivers = openvswitch
extension_drivers = port_security

[ml2_type_flat]
flat_networks = physnet1


[ml2_type_vxlan]
vni_ranges = 1:1000

$ controller> vi /etc/nova/nova.conf
[default]
...
...
use_neutron = True
linuxnet_interface_driver =
nova.network.linux_net.LinuxOVSIInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver

[neutron]
auth_url = http://controller:5000
auth_type = password
project_domain_name = default
user_domain_name = default
region_name = RegionOne
project_name = service
username = neutron
password = qwer1234
service_metadata_proxy = True
metadata_proxy_shared_secret = metadata_secret

$ controller> setsebool -P neutron_can_network on

```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ controller> setsebool -P daemons_enable_cluster_mode on
$ controller> firewall-cmd --add-port=9696/tcp --permanent
$ controller> firewall-cmd --reload
# 방화벽 및 SELinux를 설정

$ controller> ln -s /etc/neutron/plugins/ml2/ml2_conf.ini
/etc/neutron/plugin.ini
$ controller> su -s /bin/bash neutron -c "neutron-db-manage --
config-file /etc/neutron/neutron.conf --config-file
/etc/neutron/plugin.ini upgrade head"
$ controller> systemctl enable --now neutron-server neutron-
metadata-agent
$ controller> systemctl restart openstack-nova-api
# neutron DB import 및 서비스를 등록
```

- Neutron Install (network)


```
$ network> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install openstack-neutron openstack-neutron-ml2 openstack-
neutron-openvswitch libibverbs
# neutron 및 관련 모듈을 설치

$ network> vi /etc/neutron/neutron.conf
[DEFAULT]
core_plugin = ml2
service_plugins = router
auth_strategy = keystone
state_path = /var/lib/neutron
allow_overlapping_ips = True
# RabbitMQ connection info
transport_url = rabbit://openstack:qwer1234@controller

[keystone_auth_token]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = neutron
password = qwer1234

[oslo_concurrency]
lock_path = $state_path/lock

$ network> vi /etc/neutron/dhcp_agent.ini
[DEFAULT]
interface_driver = openvswitch
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = true
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ network> vi /etc/neutron/metadata_agent.ini
nova_metadata_host = controller
metadata_proxy_shared_secret = metadata_secret

[cache]
memcache_servers = controller:11211

$ network> vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,vlan,gre,vxlan
tenant_network_types = vxlan
mechanism_drivers = openvswitch
extension_drivers = port_security

[ml2_type_flat]
flat_networks = physnet1

[ml2_type_vxlan]
vni_ranges = 1:1000
# 끝에 추가


$ network> vi /etc/neutron/plugins/ml2/openvswitch_agent.ini
[securitygroup]
firewall_driver = openvswitch
enable_security_group = true
enable_ipset = true

[agent]
tunnel_types = vxlan
prevent_arp_spoofing = True

[ovs]
local_ip = 10.10.10.20
bridge_mappings = physnet1:br-eth1
# 끝에 추가
# local_ip = network Node IP

$ network> setsebool -P neutron_can_network on
$ network> setsebool -P haproxy_connect_any on
$ network> setsebool -P daemons_enable_cluster_mode on
$ network> vi ovsofctl.te
module ovsofctl 1.0;

require {
    type neutron_t;
    type neutron_exec_t;
    type neutron_t;
    type dnsmasq_t;
    class file execute_no_trans;
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

        class capability { dac_override sys_rawio };
    }

#===== neutron_t =====
allow neutron_t self:capability { dac_override sys_rawio };
allow neutron_t neutron_exec_t:file execute_no_trans;

#===== dnsmasq_t =====
allow dnsmasq_t self:capability dac_override;

$ network> checkmodule -m -M -o ovsofctl.mod ovsofctl.te
$ network> semodule_package --outfile ovsofctl.pp --module
ovsofctl.mod
$ network> semodule -i ovsofctl.pp
$ network> systemctl disable --now firewalld
# Selinux 및 방화벽을 설정

$ network> ln -s /etc/neutron/plugins/ml2/ml2_conf.ini
/etc/neutron/plugin.ini
$ network> systemctl enable --now openvswitch
$ network> ovs-vsctl add-br br-int
$ network> ovs-vsctl add-br br-eth1
$ network> ovs-vsctl add-port br-eth1 ens32
$ network> vi /etc/sysconfig/network-scripts/ifcfg-ens32
TYPE=Ethernet
BOOTPROTO=static
NAME=ens160
DEVICE=ens160
ONBOOT=yes

$ network> vi /var/tmp/network_interface.sh
#!/bin/bash

ip link set up br-eth1
ip addr add 192.168.10.20/24 dev br-eth1
route add default gw 192.168.10.2 dev br-eth1
echo "nameserver 8.8.8.8" > /etc/resolv.conf


$ network> chmod 755 /var/tmp/network_interface.sh
$ network> vi /etc/systemd/system/set_interface.service

[Unit]
Description=Description for sample script goes here
After=network.target

[Service]
Type=simple
ExecStart=/var/tmp/network_interface.sh
TimeoutStartSec=0

[Install]
WantedBy=default.target

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ systemctl enable set_interface
$ init 6
# network 인터페이스 주의 !!! ( ex : ens160 )

$ network> for service in dhcp-agent l3-agent metadata-agent
openvswitch-agent; do
systemctl enable --now neutron-$service
done
# neutron 서비스를 등록 및 시작
```

- Neutron Install (compute)

```
$ compute> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install openstack-neutron openstack-neutron-ml2 openstack-
neutron-openvswitch
# neutron 및 관련 모듈을 설치


$ compute> vi /etc/neutron/neutron.conf
[DEFAULT]
core_plugin = ml2
service_plugins = router
auth_strategy = keystone
state_path = /var/lib/neutron
allow_overlapping_ips = True
transport_url = rabbit://openstack:qwer1234@controller

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = neutron
password = qwer1234

[oslo_concurrency]
lock_path = $state_path/lock

$ compute> vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,vlan,gre,vxlan
tenant_network_types = vxlan
mechanism_drivers = openvswitch
extension_drivers = port_security

[ml2_type_flat]
flat_networks = physnet1
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
[ml2_type_vxlan]
vni_ranges = 1:1000
# 끝에 추가

$ compute> vi /etc/neutron/plugins/ml2/openvswitch_agent.ini
[securitygroup]
firewall_driver = openvswitch
enable_security_group = true
enable_ipset = true

[agent]
tunnel_types = vxlan
prevent_arp_spoofing = True


[ovs]
local_ip = 10.10.10.20
# 끝에 추가
# local_ip = compute node IP

$ compute> vi /etc/nova/nova.conf
[default]
...
...
use_neutron = True
linuxnet_interface_driver =
nova.network.linux_net.LinuxOVSIInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
vif_plugging_is_fatal = True
vif_plugging_timeout = 300

[neutron]
auth_url = http://controller:5000
auth_type = password
project_domain_name = default
user_domain_name = default
region_name = RegionOne
project_name = service
username = neutron
password = qwer1234
service_metadata_proxy = True
metadata_proxy_shared_secret = metadata_secret

$ compute> setsebool -P neutron_can_network on
$ compute> setsebool -P daemons_enable_cluster_mode on
$ compute> vi ovsofctl 1.0;

require {
    type neutron_t;
    type neutron_exec_t;
    type neutron_t;
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

    type dnsmasq_t;
    class file execute_no_trans;
    class capability { dac_override sys_rawio };
}

#===== neutron_t =====
allow neutron_t self:capability { dac_override sys_rawio };
allow neutron_t neutron_exec_t:file execute_no_trans;

#===== dnsmasq_t =====
allow dnsmasq_t self:capability dac_override;

$ network> checkmodule -m -M -o ovsofctl.mod ovsofctl.te
$ network> semodule_package --outfile ovsofctl.pp --module
ovsofctl.mod
$ network> semodule -i ovsofctl.pp
$ systemctl disable --now firewalld
# Selinux 및 방화벽을 설정

$ compute> ln -s /etc/neutron/plugins/ml2/ml2_conf.ini
/etc/neutron/plugin.ini
$ compute> systemctl enable --now openvswitch
$ compute> ovs-vsctl add-br br-int
$ compute> systemctl restart openstack-nova-compute
$ compute> systemctl enable --now neutron-openvswitch-agent
# neutron 서비스를 등록


```

- 확인

```


$ controller ~(keystone)> openstack router create router
+-----+-----+
| Field                | Value |
+-----+-----+
| admin_state_up       | UP    |
| availability_zone_hints |      |
| availability_zones    |      |
| created_at           | 2020-08-07T00:05:40Z |
| description          |      |
| distributed           | False |
| external_gateway_info | null  |
| flavor_id            | None  |
|

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| ha | False
| id | f40d6130-a01c-486a-b088-3f27c9f57607
| location | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone= |
| name | router
| project_id | c76211c24a1f460ca67274d655d46725
| revision_number | 1
| routes |
| status | ACTIVE
| tags |
| updated_at | 2020-08-07T00:05:40Z
+-----+-----+

$ controller ~(keystone)> openstack network create int --provider-
network-type vxlan
+-----+-----+
| Field | Value |
+-----+-----+
| admin_state_up | UP |
| availability_zone_hints | |
| availability_zones | |
| created_at | 2020-08-07T00:05:58Z |
| description | |
| dns_domain | None |
| id | 0edec63e-cc62-4e93-8962-d0ad2df27bc8 |
| ipv4_address_scope | None |
| ipv6_address_scope | None |
| is_default | False |
| is_vlan_transparent | None |
| location | cloud='', project.domain_id=,
project.domain_name='default',
```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```


project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone= |
| mtu | 1450
|
| name | int
|
| port_security_enabled | True
|
| project_id | c76211c24a1f460ca67274d655d46725
|
| provider:network_type | vxlan
|
| provider:physical_network | None
|
| provider:segmentation_id | 1
|
| qos_policy_id | None
|
| revision_number | 1
|
| router:external | Internal
|
| segments | None
|
| shared | False
|
| status | ACTIVE
|
| subnets |
|
| tags |
|
| updated_at | 2020-08-07T00:05:58Z
|
+-----+-----+

$ controller ~(keystone)> openstack subnet create int-sub --network
int \
--subnet-range 1.1.1.0/24 --gateway 1.1.1.1 \
--dns-nameserver 8.8.8.8
+-----+-----+
| Field | Value |
|
+-----+-----+
| allocation_pools | 1.1.1.2-1.1.1.254 |
| cidr | 1.1.1.0/24 |
| created_at | 2020-08-07T00:06:25Z |
| description | |
| dns_nameservers | 8.8.8.8 |
|


```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


```
| dns_publish_fixed_ip | None
| enable_dhcp          | True
| gateway_ip           | 1.1.1.1
| host_routes          |
| id                   | 800bc5af-45e9-4719-8969-4c154bc111d6
| ip_version           | 4
| ipv6_address_mode    | None
| ipv6_ra_mode         | None
| location              | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone=
| name                 | int-sub
| network_id           | 0edec63e-cc62-4e93-8962-d0ad2df27bc8
| prefix_length        | None
| project_id           | c76211c24a1f460ca67274d655d46725
| revision_number       | 0
| segment_id           | None
| service_types        |
| subnetpool_id        | None
| tags                 |
| updated_at           | 2020-08-07T00:06:25Z
|
+-----+-----+
$ controller ~(keystone)> openstack router add subnet router int-sub
$ controller ~(keystone)> openstack network create \
--provider-physical-network physnet1 \
--provider-network-type flat --external ext
+-----+-----+
| Field                | Value
|
+-----+-----+
| admin_state_up       | UP
|
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| availability_zone_hints |
| availability_zones |
| created_at | 2020-08-07T00:06:47Z
| description |
| dns_domain | None
| id | 68e5adb0-a8c4-473b-88a9-fdaaf6f12ec2
| ipv4_address_scope | None
| ipv6_address_scope | None
| is_default | False
| is_vlan_transparent | None
| location | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone= |
| mtu | 1500
| name | ext
| port_security_enabled | True
| project_id | c76211c24a1f460ca67274d655d46725
| provider:network_type | flat
| provider:physical_network | physnet1
| provider:segmentation_id | None
| qos_policy_id | None
| revision_number | 1
| router:external | External
| segments | None
| shared | False
| status | ACTIVE
| subnets |
| tags |
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| updated_at          | 2020-08-07T00:06:47Z
|
+-----+-----+
$ controller ~(keystone)> openstack subnet create ext-sub \
--network ext --subnet-range 192.168.10.0/24 \
--allocation-pool start=192.168.10.150,end=192.168.10.200 \
--gateway 192.168.10.2 --dns-nameserver 8.8.8.8
+-----+-----+
| Field                | Value
|
+-----+-----+
| allocation_pools      | 192.168.10.150-192.168.10.200
| cidr                  | 192.168.10.0/24
| created_at           | 2020-08-07T00:07:21Z
| description           |
| dns_nameservers       | 8.8.8.8
| dns_publish_fixed_ip | None
| enable_dhcp           | True
| gateway_ip           | 192.168.10.2
| host_routes           |
| id                   | 31a92331-f102-4c4e-8c02-f97baa9eab28
| ip_version            | 4
| ipv6_address_mode     | None
| ipv6_ra_mode          | None
| location              | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone=
| name                  | ext-sub
| network_id            | 68e5adb0-a8c4-473b-88a9-fdaaf6f12ec2
| prefix_length         | None
| project_id            | c76211c24a1f460ca67274d655d46725
| revision_number       | 0
| segment_id            | None
|
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


```
| service_types      |
| subnetpool_id     | None
| tags              |
| updated_at        | 2020-08-07T00:07:21Z
|
+-----+-----+

$ controller ~(keystone)> openstack router set router --external-
gateway ext

$ controller ~(keystone)> openstack network rbac list
+-----+-----+-----+-----+
| ID                                     | Object Type | Object ID
|
+-----+-----+-----+-----+
| 4e8ebe0b-60f0-485c-8696-74378068c844 | network     | 68e5adb0-a8c4-
473b-88a9-fdaaf6f12ec2 |
+-----+-----+-----+-----+


$ controller ~(keystone)> wget http://cloud-
images.ubuntu.com/releases/18.04/release/ubuntu-18.04-server-
cloudimg-amd64.img -P /var/kvm/images
$ controller ~(keystone)> openstack image create "Ubuntu1804" --file
/var/kvm/images/ubuntu-18.04-server-cloudimg-amd64.img --disk-format
qcow2 --container-format bare --public
# Ubuntu18.04 이미지를 다운로드 후, 등록

$ controller ~(keystone)> openstack security group create all-port
+-----+-----+-----+-----+
| Field          | Value
|
+-----+-----+-----+-----+
| created_at     | 2020-08-07T00:10:31Z
|
| description    | all-port
|
| id             | 97224218-b304-4076-9645-d68092a9366a
|
| location       | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone=
| name           | all-port
|
| project_id     | c76211c24a1f460ca67274d655d46725
|
| revision_number | 1
|
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| rules | created_at='2020-08-07T00:10:32Z',
direction='egress', ethertype='IPv6', id='333de7e9-5c1b-4b2f-bb0e-
2dalb878abb6', updated_at='2020-08-07T00:10:32Z' |
| | created_at='2020-08-07T00:10:32Z',
direction='egress', ethertype='IPv4', id='644e18e1-4f4e-42ad-bef8-
937e47254a27', updated_at='2020-08-07T00:10:32Z' |
| stateful | True
|
| tags | []
|
| updated_at | 2020-08-07T00:10:32Z
|
+-----+-----+


$ controller ~(keystone)> openstack security group rule create --
protocol icmp --ingress all-port
+-----+-----+
| Field | Value |
|
+-----+-----+
| created_at | 2020-08-07T00:13:31Z |
|
| description | |
|
| direction | ingress |
|
| ether_type | IPv4 |
|
| id | 27688481-047b-4fc0-948c-de109e46d7f5 |
|
| location | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone= |
| name | None |
|
| port_range_max | None |
|
| port_range_min | None |
|
| project_id | c76211c24a1f460ca67274d655d46725 |
|
| protocol | icmp |
|
| remote_group_id | None |
|
| remote_ip_prefix | 0.0.0.0/0 |
|
| revision_number | 0 |
|
| security_group_id | 97224218-b304-4076-9645-d68092a9366a |
|
| tags | [] |
|
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| updated_at      | 2020-08-07T00:13:31Z
|
+-----+-----+
$ controller ~(keystone)> openstack security group rule create --
protocol tcp --dst-port 22:22 all-port
+-----+-----+
| Field          | Value
|
+-----+-----+
| created_at     | 2020-08-07T00:13:36Z
| description    |
| direction     | ingress
| ether_type     | IPv4
| id             | da2afd20-818a-4bfe-9017-c837b2bf30ec
| location       | cloud='', project.domain_id=,
project.domain_name='default',
project.id='c76211c24a1f460ca67274d655d46725', project.name='admin',
region_name='', zone=
| name           | None
| port_range_max | 22
| port_range_min | 22
| project_id     | c76211c24a1f460ca67274d655d46725
| protocol       | tcp
| remote_group_id | None
| remote_ip_prefix | 0.0.0.0/0
| revision_number | 0
| security_group_id | 97224218-b304-4076-9645-d68092a9366a
| tags           | []
| updated_at     | 2020-08-07T00:13:36Z
|
+-----+-----+
$ controller ~(keystone)> ssh-keygen -q -N ""
$ controller ~(keystone)> openstack keypair create --public-key
~/.ssh/id_rsa.pub MyKey
+-----+-----+
| Field          | Value
|
+-----+-----+


```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| fingerprint | a3:8f:44:f6:e1:4e:da:a0:90:f1:5d:dc:6a:8b:ad:76 |
| name        | MyKey |
| user_id     | 57ce8f772e374a7c9282f2674fda1ba7 |
+-----+-----+

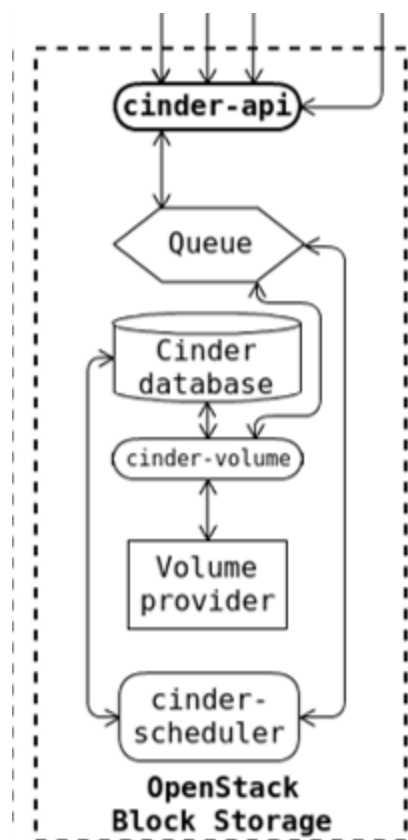
$ controller ~(keystone)> openstack flavor create --ram 1024 --disk
10 --vcpus 1 m1.small
+-----+-----+
+
| Field | Value |
+-----+-----+
+
| OS-FLV-DISABLED:disabled | False |
| OS-FLV-EXT-DATA:ephemeral | 0 |
| disk | 10 |
| id | dabfebd4-cd05-4cec-9567-78b8c9e3d6b6 |
| name | m1.small |
| os-flavor-access:is_public | True |
| properties | |
| ram | 1024 |
| rxtx_factor | 1.0 |
| swap | |
| vcpus | 1 |
+-----+-----+
+

$ controller ~(keystone)> openstack server create --image Ubuntu1804
--flavor m1.small --key Mykey --network int --security-group all-
port Ubuntu
$ controller ~(keystone)> openstack floating ip create ext
+-----+-----+
+
| Field | Value |
+-----+-----+
+
| created_at | 2020-08-07T00:16:15Z |
| description | |
| dns_domain | None |
| dns_name | None |
| fixed_ip_address | None |
| floating_ip_address | 192.168.10.191 |
| floating_network_id | 68e5adb0-a8c4-473b-88a9-fdaaf6f12ec2 |
| id | 409a4724-1e13-4150-a2e1-6b3a205c4ff6 |
| location | Munch({'cloud': '', 'region_name': '', 'zone':
None, 'project': Munch({'id': 'c76211c24a1f460ca67274d655d46725',
'name': 'admin', 'domain_id': None, 'domain_name': 'default'})}) |
```



	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- keystone으로부터 사용자의 인증이 완료되면 DB를 읽어 사용자 사용이 가능한 리소스를 생성, 혹은 할당하는 역할을 수행한다.

Ciner의 논리 아키텍처



구성요소	역할
Cinder-api	요청에 따라 storage를 할당, 확장하는 기능을 수행
Queue	각 구성요소 간의 통신기능을 수행
Cinder database	Cinder 서비스를 수행하기 위한 일련의 정보들은 보관, 관리하는 DB
cinder voulme	Cinder 서비스를 통해 가상화되어진 Storage voulme, voulme을 관리 및 업데이트
volume provider	storage volume을 생성, 확장하는 서비스
cinder scheduler	요청이 다수인 경우 큐를 통해 받은 메시지를 수행

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- Nova에서 생성된 인스턴스에서 확장해서 사용할 수 있는 저장 공간을 생성, 삭제하고 인스턴스에 연결 할 수 있는 기능을 제공
- Cinder은 물리 하드 디스크를 LVM(Logical Volume Manager)으로 설정
- 설정된 LVM은 Cinder-conf와 nova.conf의 환경을 설정해서 cinder-volume을 할당가능
- cinder-api로 생성된 볼륨은 단일 인스턴스 또는 여러 인스턴스에 할당이 가능

Ciner driver type


- Cinder의 기본 블록 스토리지 드라이버는 ISCSI 기반의 LVM으로 LVM이란 하드 디스크를 파티션 대신 논리 볼륨으로 할당하고, 디스크 여러 개를 보다 효율적이고 유연하게 관리 할 수 있는 방식을 의미
- 블록 장치는 물리 볼륨으로 초기화해야 하며, 논리 볼륨으로 생성하려면 물리 볼륨을 볼륨 그룹으로 통합수행이 요구됨

Ciner의 Install

- **Cinder User, Service, DB 생성**

```
$ controller ~(keystone)> openstack user create --domain default --project service --password qwer1234 cinder
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| default_project_id | 7c10c02365be496fb47f12bfd40fe4a7 |
| domain_id        | default                               |
| enabled          | True                                 |
| id               | 1f9dbcbb529a45c28b5bb8b035ea277a |
| name             | cinder                              |
| options          | {}                                  |
| password_expires_at | None                               |
+-----+-----+

$ controller ~(keystone)> openstack role add --project service --user cinder admin
```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ controller ~(keystone)> openstack service create --name cinderv3 -
-description "OpenStack Block Storage" volumev3
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| description | OpenStack Block Storage                 |
| enabled     | True                                    |
| id          | 225ceadb699d4e79adf30769cd872fef       |
| name        | cinderv3                               |
| type        | volumev3                               |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne volumev3 public http://controller:8776/v3/%(tenant_id)s
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                    |
| id          | 6bf917232caa43eab3b83959fb19cb45       |
| interface   | public                                 |
| region      | RegionOne                             |
| region_id   | RegionOne                             |
| service_id  | 225ceadb699d4e79adf30769cd872fef       |
| service_name | cinderv3                               |
| service_type | volumev3                               |
| url         | http://controller:8776/v3/%(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne volumev3 internal
http://controller:8776/v3/%(tenant_id)s
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                    |
| id          | c5987fc3d9eb4fb79a2e8cf73a274936       |
| interface   | internal                               |
| region      | RegionOne                             |
| region_id   | RegionOne                             |
| service_id  | 225ceadb699d4e79adf30769cd872fef       |
| service_name | cinderv3                               |
| service_type | volumev3                               |
| url         | http://controller:8776/v3/%(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne volumev3 admin http://controller:8776/v3/%(tenant_id)s
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                    |
| id          | eff2398584944c0fa7575d1991d725fe       |
| interface   | admin                                 |
| region      | RegionOne                             |
+-----+-----+
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
| region_id      | RegionOne                                     |
| service_id     | 225ceadb699d4e79adf30769cd872fef             |
| service_name   | cinderv3                                       |
| service_type   | volumev3                                       |
| url            | http://controller:8776/v3/(tenant_id)s      |
+-----+-----+
$ controller> mysql -u root -p
$ MariaDB> create database cinder;
$ MariaDB> grant all privileges on cinder.* to cinder@'localhost'
identified by 'qwer1234';
$ MariaDB> grant all privileges on cinder.* to cinder@'%' identified
by 'qwer1234';
$ MariaDB> flush privileges;
$ MariaDB> exit;
```

- Cinder Install (controller)


```
$ controller> dnf --enablerepo=centos-openstack-
ussuri,PowerTools,epel -y install openstack-cinder
# cinder 및 관련 모듈을 설치

$ controller> vi /etc/cinder/cinder.conf
[DEFAULT]
my_ip = controller
log_dir = /var/log/cinder
state_path = /var/lib/cinder
auth_strategy = keystone
transport_url = rabbit://openstack:qwer1234@controller
enable_v3_api = True

[database]
connection = mysql+pymysql://cinder:qwer1234@controller/cinder

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = cinder
password = qwer1234

[oslo_concurrency]
lock_path = $state_path/tmp
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ controller> su -s /bin/bash cinder -c "cinder-manage db sync"
$ controller> systemctl enable --now openstack-cinder-api openstack-
cinder-scheduler
# cinder DB를 Import 및 서비스를 등록

$ controller> echo "export OS_VOLUME_API_VERSION=3" >> ~/admin_key
$ controller> source ~/admin_key
# key 파일을 수정합니다.

$ controller> firewall-cmd --add-port=8776/tcp --permanent
$ controller> firewall-cmd --reload
# 방화벽을 설정
```

- Cinder Install (compute)

```
$ compute> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install openstack-cinder targetcli
# cinder 및 관련 모듈을 설치


$ compute> fdisk ...
# LVM의 타입으로 파티션을 추가
# cinder 이름으로 vg를 생성

$ controller> scp /etc/cinder/cinder.conf
compute:/etc/cinder/cinder.conf
$ compute> vi /etc/cinder/cinder.conf
[default]
my_ip = compute
...
...
glance_api_servers = http://controller:9292
enabled_backends = lvm

[lvm]
target_helper = lioadm
target_protocol = iscsi
target_ip_address = compute
volume_group = cinder
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volumes_dir = $state_path/volumes

$ compute> vi /etc/nova/nova.conf
[cinder]
os_region_name = RegionOne

$ compute> systemctl restart openstack-nova-compute
$ compute> systemctl enable --now openstack-cinder-volume
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ compute> vi iscsiadm.te
module iscsiadm 1.0;

require {
    type iscsid_t;
    class capability dac_override;
}


#===== iscsid_t =====
allow iscsid_t self:capability dac_override;

$ compute> checkmodule -m -M -o iscsiadm.mod iscsiadm.te
$ compute> semodule_package --outfile iscsiadm.pp --module
iscsiadm.mod
$ compute> semodule -i iscsiadm.pp
$ compute> firewall-cmd --add-service=iscsi-target --permanent
$ compute> firewall-cmd --reload
# SELinux 및 방화벽을 설정
```

- Cinder 확인

```
$ controller ~/(keystone)> openstack volume service list
+-----+-----+-----+-----+-----+-----+
| Binary          | Host          | Zone | Status | State | Updated At |
+-----+-----+-----+-----+-----+-----+
| cinder-scheduler | controller    | nova | enabled | up    | 2020-08-07T01:29:22.000000 |
| cinder-volume    | compute@lvm  | nova | enabled | up    | 2020-08-07T01:29:22.000000 |
+-----+-----+-----+-----+-----+-----+

$ controller ~/(keystone)> openstack volume create --size 1 test
+-----+-----+-----+-----+-----+-----+
| Field          | Value |
+-----+-----+-----+-----+-----+
| attachments    | []    |
| availability_zone | nova  |
| bootable       | false |
| consistencygroup_id | None  |
| created_at     | 2020-08-07T01:46:06.000000 |
| description    | None  |
| encrypted      | False |
| id             | aa07bf85-424d-478c-ae52-648ddc588465 |
| migration_status | None  |
| multiattach    | False |
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```


| name           | test           |
| properties     |                |
| replication_status | None          |
| size           | 1              |
| snapshot_id    | None           |
| source_volid   | None           |
| status         | creating       |
| type           | __DEFAULT__   |
| updated_at     | None           |
| user_id        | 57ce8f772e374a7c9282f2674fda1ba7 |
+-----+-----+
$ controller ~/ (keystone)> openstack volume list
+-----+-----+-----+-----+-----+
| ID | Name | Status | Size |
+-----+-----+-----+-----+
| aa07bf85-424d-478c-ae52-648ddc588465 | test | available | 1 |
+-----+-----+-----+-----+

```

- Cinder backup service (수정 중)

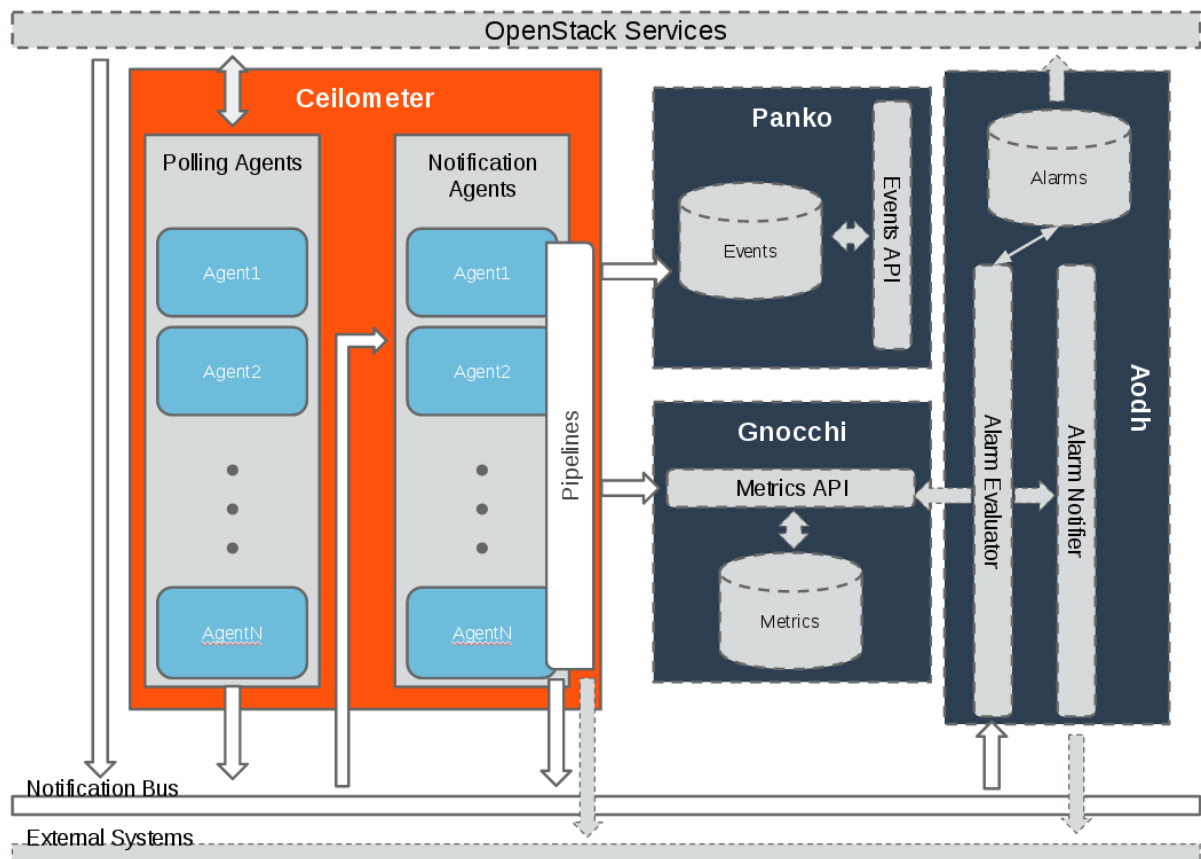
리소스의 사용량과 부하를 관리하는 서비스 : Ceilometer

- Ceilometer은 각 서비스들의 예상 부하에 따라 추가 작업과 노드를 관리하는 역할을 수행
- 클라우드에서 배포된 자원의 사용량과 성능을 측정해 사용자가 자원 상태를 모니터링 할 수 있는 기능을 제공

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


- Ceilometer는 리버티 버전에서 기존에 알람 서비스를 하던 aodh로 분리

Ceilometer의 구성요소



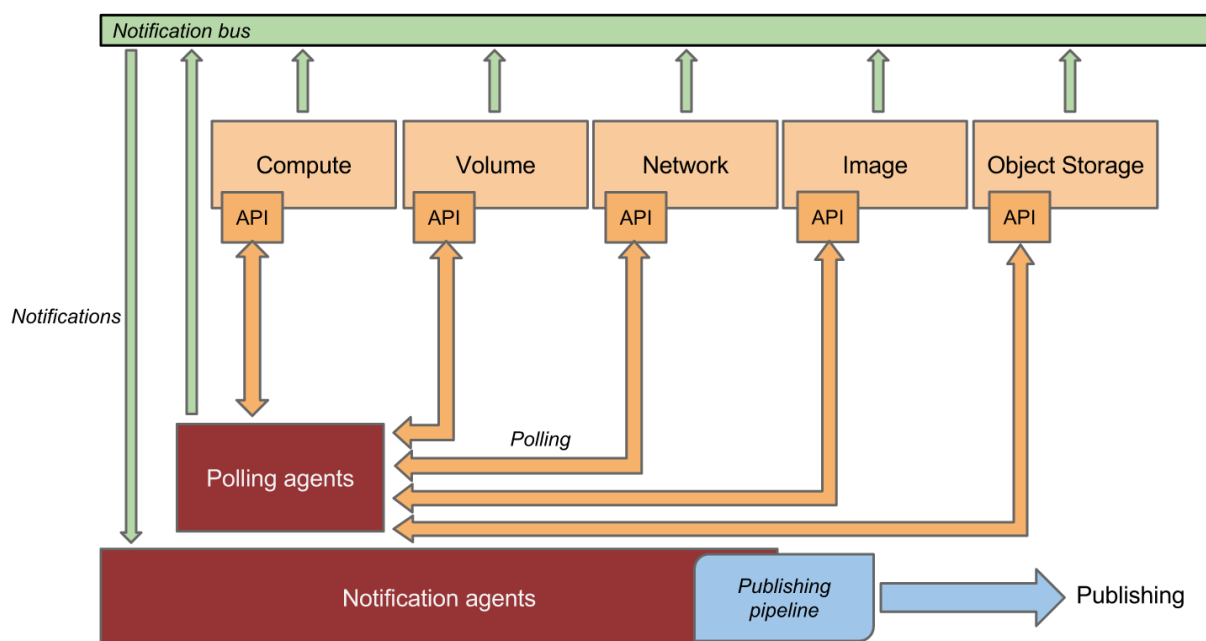
핵심 서비스	역할
Polling agent	OpenStack 서비스를 폴링하고 미터를 빌드하도록 설계된 데몬 프로세스
Notification agent	메시지 큐에서 알람을 듣고 이벤트 및 샘플로 변환하며 파이프 라인 조치를 적용하도록 설계된 데몬 프로세스

- Ceilometer로 표준화 및 수집 된 데이터는 다양한 대상으로 전송
- Gnocchi는 이러한 스토리지 및 쿼리를 최적화하기 위해 시계열 형식으로 측정 데이터를 캡처하도록 개발되어짐
- Aodh는 사용자의 정의 규칙이 깨졌을 때 경고를 보낼 수 있는 경보 서비스를 의미

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


- Ceilometer은 이와 같이 리소스를 감시 및 예상하여 다른 작업을 수행 할 수 있도록 하는 서비스를 의미

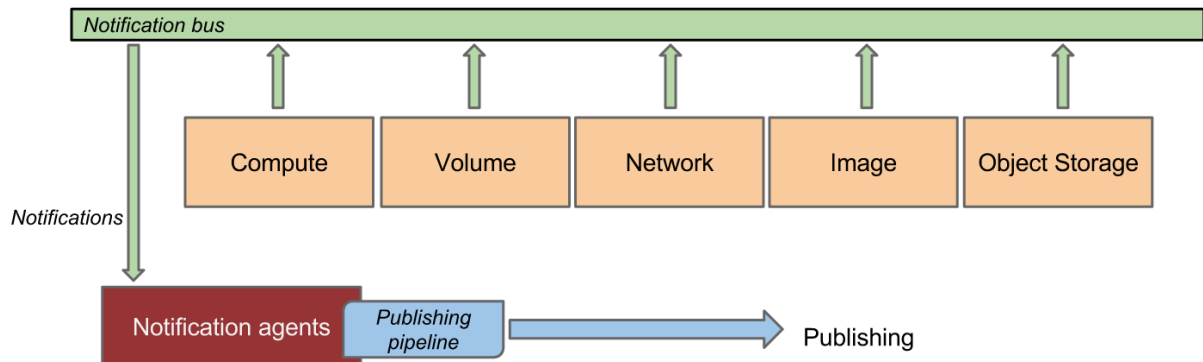
Ceilomter의 데이터 수집



- 위의 그림과 같이 Polling agents에서 각 서비스들의 API의 리소스를 읽어 데이터를 수집
- Notification agents는 Polling agents에서 읽힌 데이터들을 토대로 Ceilometer 서비스 혹은 Events로 변환하는 역할을 수행

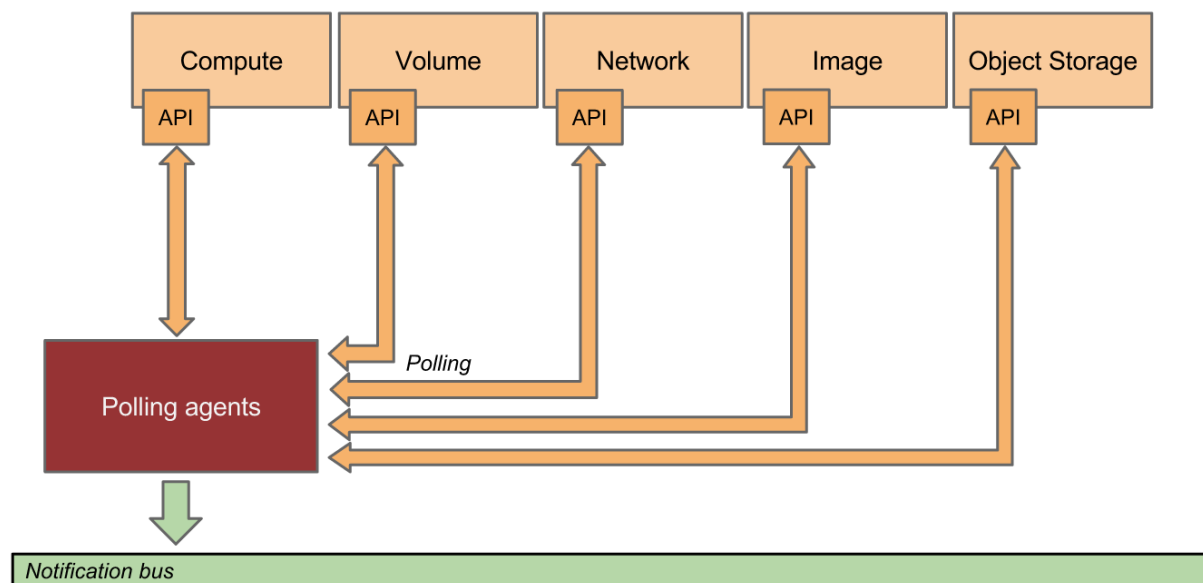
Ceilomter의 데이터 처리

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	




- 수집된 데이터는 위의 그림처럼 Notification bus를 통해 엔드포인트로 재분배하여이벤트 및 샘플로 처리하는 역할을 수행

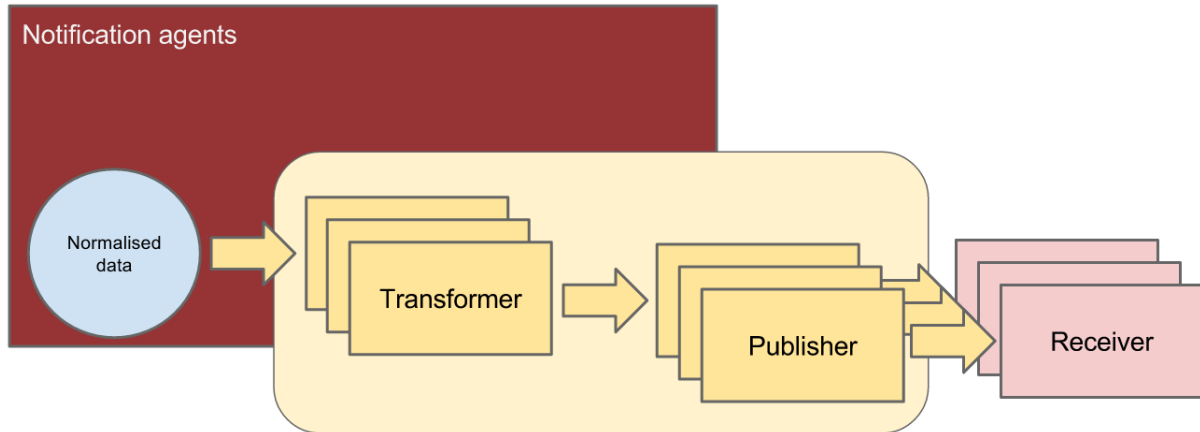
Ceilometer의 데이터 요청



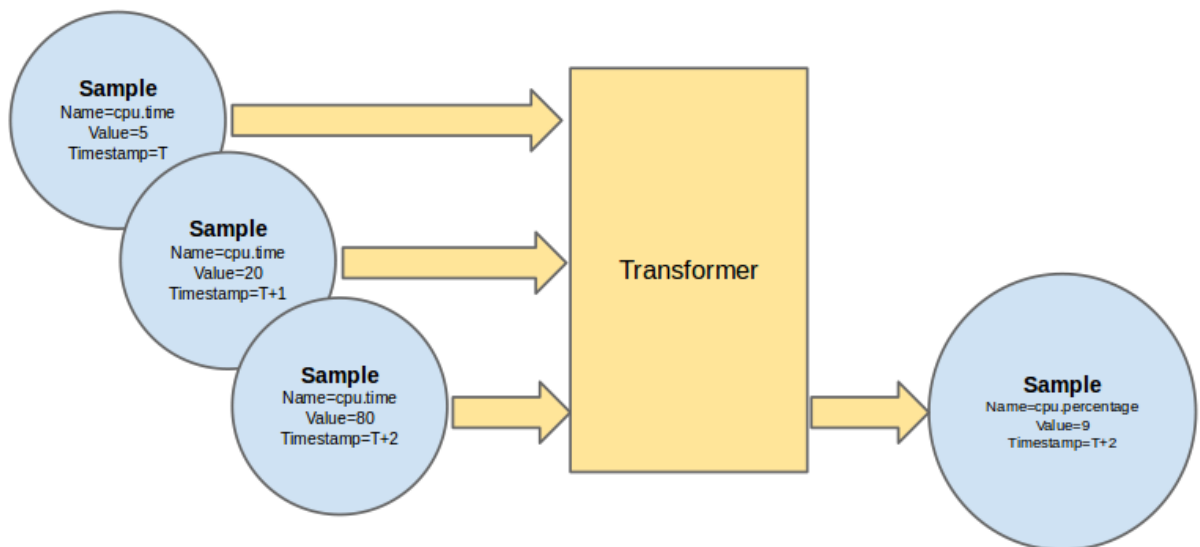
- 데이터 요청은 수집된 자료들을 토대로 서로 데이터를 주고 받으며, Polling agents라는 클라우드 컨트롤러에서 처리되며, 여러 플러그인을 사용하여 데이터와 통신

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


Ceilometer의 데이터 처리 및 변형



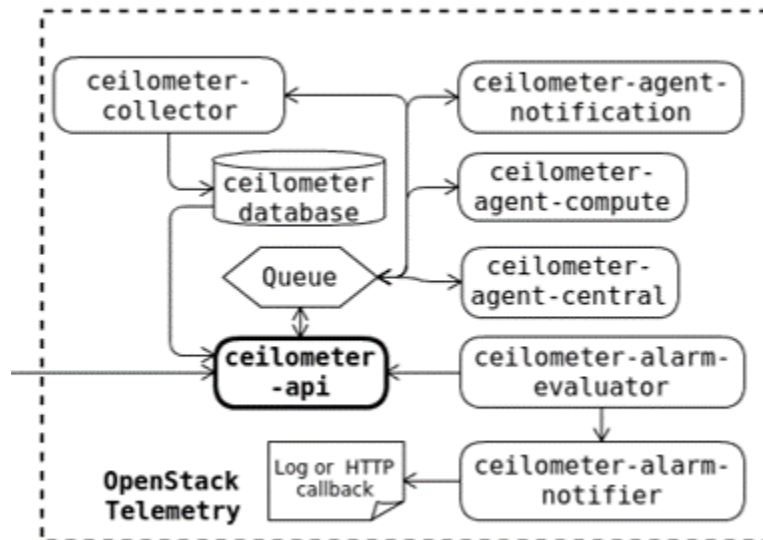
Pipeline: a set of transformers mutating data points into something that publishers know how to send to external systems.



- 위의 그림은 같이 수집된 데이터를 수집, 분석 및 변형하여 배포하는 과정을 나타낸 그림으로 Ceilometer은 각 서비스들의 데이터를 수집하여, 변형시키는 여러 소스를 제공


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Ceilometer의 아키텍처



구성요소	역할
Ceilometer-collector	중앙 관리서버에서 실행되며, Queue 모니터링 하는 서비스
Ceilometer-agent-notification	ceilometer-collector와 함께 중앙 관리서버에서 실행, Queue를 이용해 이벤트와 미러링 데이터를 기록
Ceilometer-agent-compute	각 컴퓨팅 노드에 설치해서 실행, 자원 활용 통계로 사용
Ceilometer-agent-central	중앙 관리 서버에서 실행, 인스턴스에 연결되지 않은 자원이나 컴퓨터 노드의 활용 가능한 자원 통계를 폴링
Ceilometer-alarms-evaluator	하나 이상의 중앙 관리 서버에서 실행, 슬라이딩 시간대에 임계 값을 추가할 때 발생하는 경보 시점을 결정
Ceilometer-alarms-notifier	하나 이상의 중앙 관리 서버에서 실행되며, 샘플을 수집하는 임계 값 평가에 따라 알람을 설정
Ceilometer database	수집한 데이터를 저장할 Ceilometer 데이터 베이스
Ceilometer-api	하나 또는 그 이상의 중앙 관리 서버에서 실행되며 데이터베이스에서 데이터 액세스를 제공

- Ceilometer은 OpenStack 내에서 각 리소스들을 관리하고 이벤트, 알람(현재는 분리) 등의 인터페이스를 제공하는 역할을 수행하는 서비스를 의미한다.
- OpenStack 내에서의 Ceilometer은 각 서비스각의 이벤트, 사용량 등의 대한 종합적인 데이터를 수집하여 이를 새로운 데이터를 창출하여, 사용자의 편의에 따라, 혹은 서로 다른

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

서비스를 묶어 사용할 수 있는 관리 서비스의 역할을 수행한다.

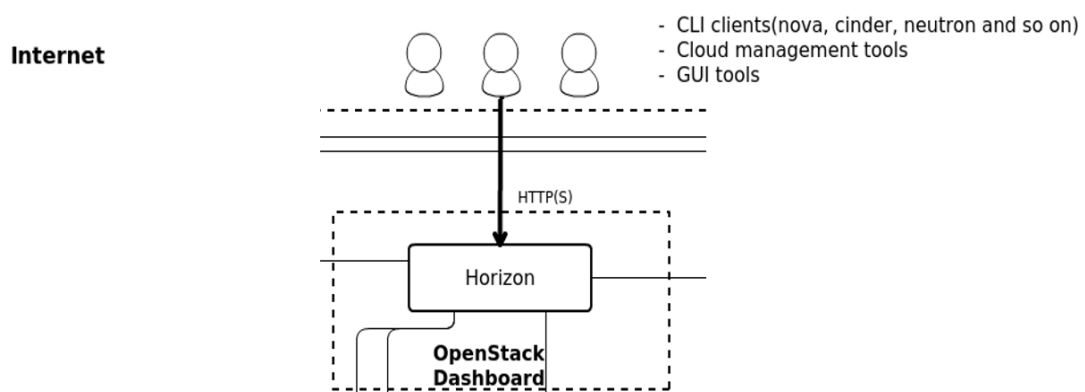
Ceilometer Install

- Ceilometer는 주로 Compute 서비스인 NOVA와 함께 설치되며 설정되었으나 현재는 독립하여 자체적인 설정파일이 존재
- 하지만 현재도 설치와 설정은 편의를 위해 Nova와 함께 되어지고 있음.


외부 인터페이스 대시보드 서비스 : Horizon

- 사용자가 웹 User Interface를 통해 인스턴스를 생성, 삭제, 관리 등의 작업을 보다 원활하게 사용하기 위해 구현된 서비스
- Horizon은 Apache, Python, Django 프레임워크로 구현되어있다.

Horizon의 아키텍처



- Horizon은 단순히 Horizon 자체 모듈만이 존재하며, 각 설정파일에 따라 서비스들을 웹

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

상에서 확인하고, 사용할 수 있도록 자체적으로 설정이 되어있음.

- 모든 서비스의 API와 연동하여 사용자에게 웹 서비스를 제공

Horizon Install

- Horizon은 controller에서만 설치를 진행

```
$ controller> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel -y install openstack-dashboard

$ controller> vi /etc/openstack-dashboard/local_settings
ALLOWED_HOSTS = ['*', '']
# 모든 host 의 접속이 가능하게 설정


CACHES = {
    'default': {
        'BACKEND':
'django.core.cache.backends.memcached.MemcachedCache',
        'LOCATION': 'controller:11211',
    },
}

SESSION_ENGINE = "django.contrib.sessions.backends.cache"
OPENSTACK_HOST = "controller"
OPENSTACK_KEYSTONE_URL = "http://controller:5000/v3"
# openstack host 와 SESSION 서버의 host 를 지정

TIME_ZONE = "Asia/Seoul"
# 시간을 지정

WEBROOT = '/dashboard/'
LOGIN_URL = '/dashboard/auth/login/'
LOGOUT_URL = '/dashboard/auth/logout/'
LOGIN_REDIRECT_URL = '/dashboard/'
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = 'Default'
OPENSTACK_API_VERSIONS = {
    "identity": 3,
    "volume": 3,
    "compute": 2,
}
# 끝에 추가

$ controller> vi /etc/httpd/conf.d/openstack-dashboard.conf
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

....
....
WSGIApplicationGroup %{GLOBAL}
# 상단에 추가

$ controller> systemctl restart httpd
# httpd 를 재시작

$ controller> setsebool -P httpd_can_network_connect on
$ controller> firewall-cmd --add-service={http,https} --permanent
$ controller> firewall-cmd --reload
# SELinux 및 방화벽을 설정

```


- 접속확인

```
http://[ controller 의 IP ]/dashboard/
```

오브젝트 스토리지 관리 서비스 : Swift


- Swift는 다른 서비스와는 다르게 단독으로 구성되며, 클라우드 스토리지 서비스와 거의 동일한 역할을 수행
- Swift는 Object Storage 서비스 중 하나로 분산 구조의 Object 데이터의 저장 스토리지 체계로 구성
- 빠른 성능 및 대용량 저장공간이 필요할 때 사용
- 동영상, 이미지, 디스크 등의 대용량, 비정형 데이터를 파일과 메타데이터로 저장하여 분산 관리를 진행

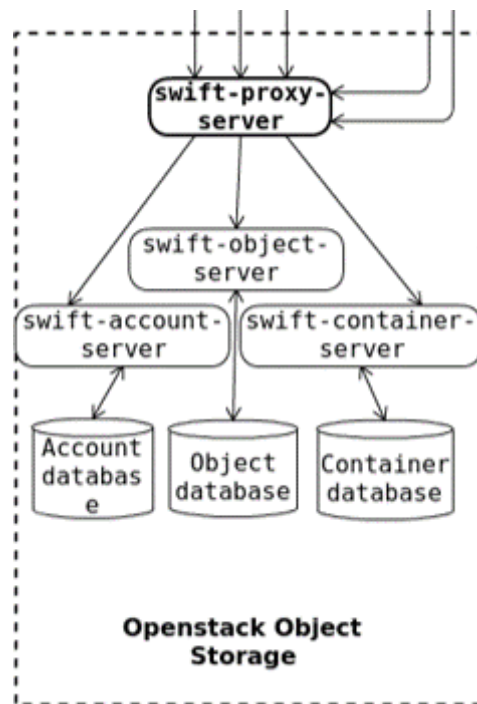
Swift의 구성요소

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- Swift는 스토리지 공간 여러 개를 합쳐 하나의 커다란 공간으로 가상화하고, 그 안에서 사용자만의 별도 스토리지 공간이 있는 것처럼 다시 가상화
- Swift-proxy-server는 다수의 스토리지 노드르 관리하며 사용자 인증을 담당, 최근에는 keystone에서 인증을 처리하며, 프록시 서버와 함께 설치하는 추세
- 기본적으로 스토리지 노드에는 swift-account-server, swift, compute-server, swift-object-server가 실행되며 실제 메타데이터파일이나 오브젝트에 해당하는 데이터 파일을 저장
- Swift 역시도 Nova를 구성할 때와 마찬가지로 스토리지 노드가 여러 호스트로 구성이 가능
- 각 스토리지 노드에는 Swift-account-server, Swift-container-server, Swift-object-server가 실행
- 서버들은 관리자가 설정한 해당 포트로 서로 통신
- 스토리지 노드 중 하나라도 손상이 되면 데이터를 잃지 않도록 데이터 복제(Replica) 프로세스가 함께 실행

Swift의 아키텍처

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	




구성요소	역할
swift-proxy-server	사용자가 서비스를 다루기 위한 REST API 서비스
swift-account-server	계정 정보 및 사용자 정보를 관리하고 각 컨테이너 별 정보를 관리하기 위한 데몬 프로세스
swift-container-server	사용자 계정의 컨테이너를 관리하는 서비스
swift-object-server	실제 데이터를 저장하고 관리하는 서비스

- 어카운트, 컨테이너는 별도의 메타데이터가 데이터베이스로 관리
- 오브젝트는 저장 공간에 직접 저장되는 방식으로 구성
- swift-proxy-server는 OpenStack의 Object API를 제공하는 역할을 수행
- 사용자는 API를 사용해 데이터를 사용

Swift Ring

- 스토리지 파일은 자신이 관리하는 데이터를 서로 공유하고 Ring인 파일이 어느 노드에,

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

어떤 데이터가 들어 있는 지를 인지하는 역할을 수행


- Ring 파일은 프록시 노드에서 생성해 모든 스토리지 노드가 동일하게 소유
- **Ring 파일의 구성요소**
 - 디바이스 정보
 - 존(Zone) 번호
 - 스토리지 노드 IP
 - 포트
 - 디바이스 명
 - 디바이스 비중
 - 기타 디바이스 정보

Swift의 데이터 관리

- Swift는 사용자 계정을 관리하는 어카운트, 디렉터리 개념의 컨테이너, 실제 파일을 표현하는 오브젝트로 구성
- Swift는 어카운트가 컨테이너를 포함하고, 컨테이너가 오브젝트를 포함하도록 관리
- 기본적으로 Swift에서는 프록시 노드 한 대에 스토리지 노드 다섯 대를 구성하기를 권장
- 프록시 노드들은 로드밸런서로 묶여 있어 사용자는 특정 URL 하나만 호출해도 스토리지 서비스를 자유롭게 사용가능
- 파일을 올릴 때는 설정된 리플리카로 여러 스토리지 | 노드로 분산해서 저장, 다운로드 시 그 중 한 곳을 사용


Swift와 Keystone

- Swift에는 SwAuth를 이용하는 인증 방법과 Keystone을 사용
- 최근에는 Keystone을 이용해서 주로 인증을 진행하며, Keystone에는 프로젝트, 사용자,

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Role이 존재

- 관리자(admin, swiftoperator)는 사용자 컨테이너를 생성 및 삭제 가능
- 관리자는 오브젝트도 Pull, Push, Delete 등이 가능
- 일반 사용자(Member)는 사용자와 컨테이너를 생성이 불가능
- 일반 사용자는 관리자가 미리 생성해서 권한을 소유한 컨테이너만 사용이 가능
- 일반 사용자는 관리자가 설정한 권한을 통해 오브젝트 목록을 확인 가능
- 일반 사용자는 관리자가 설정한 권한으로 데이터를 다룰 수 있음
- 특정 사용자에게 관리자(admin) 권한을 부여하려면 리셀러어드민(ResellerAdmin) 롤을 주어야 가능
- 해당 사용자는 관리자가 할 수 있는 모든 기능을 사용가능

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Swift의 이레이저 코딩기능과 스토리지 정책

- 스토리지 저장 공간을 보다 효율적으로 관리하는 것이 이레이저 코딩(Eraure Coding)
- 다양한 물리 스토리지 디바이스를 정책별로 사용할 수 있게 지원하는 스토리지 정책이 존재
- 이레이저 코딩은 HDFS, RAID 외의 스토리지에서 데이터 저장 공간의 효율성을 높이려고 설계된 데이터 복제 방식을 의미
- 이레이저 코딩은 이레이저 코드를 사용해 데이터를 인코딩하고, 데이터가 손실되면 디코딩 과정을 거쳐 원본 데이터를 복구하는 역할을 수행


Swift Install

- **Swift User, Serive 생성**

```
$ controller ~(keystone)> openstack user create --domain default --project service --password qwer1234 swift
+-----+
| Field          | Value                                     |
+-----+-----+
| default_project_id | b470c69e28db47cdbfc81e06cc67f627 |
| domain_id        | default                               |
| enabled          | True                                 |
| id               | dd2f0225406249b195e4feff91eca393 |
| name             | swift                               |
| options          | {}                                  |
| password_expires_at | None                               |
+-----+-----+

$ controller ~(keystone)> openstack role add --project service --user swift admin

$ controller ~(keystone)> openstack service create --name swift --description "OpenStack Object Storage" object-store
+-----+
| Field          | Value                                     |
+-----+-----+
| description    | OpenStack Object Storage               |
| enabled        | True                                   |
| id             | d9d7bc4b99774d3ba701e2eae93edfe2 |
| name           | swift                                 |
+-----+-----+
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```


| type          | object-store          |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne object-store public
http://network:8080/v1/AUTH_%(tenant_id)s
+-----+-----+
| Field          | Value                  |
+-----+-----+
| enabled        | True                   |
| id             | a70e1ac16a9144529ea49132cd7dd39e |
| interface      | public                 |
| region         | RegionOne              |
| region_id      | RegionOne              |
| service_id     | d9d7bc4b99774d3ba701e2eae93edfe2 |
| service_name   | swift                  |
| service_type   | object-store           |
| url            | http://network:8080/v1/AUTH_%(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne object-store internal
http://network:8080/v1/AUTH_%(tenant_id)s
+-----+-----+
| Field          | Value                  |
+-----+-----+
| enabled        | True                   |
| id             | 6b5ea7b028f94035aef5601cf35d3a29 |
| interface      | internal               |
| region         | RegionOne              |
| region_id      | RegionOne              |
| service_id     | d9d7bc4b99774d3ba701e2eae93edfe2 |
| service_name   | swift                  |
| service_type   | object-store           |
| url            | http://network:8080/v1/AUTH_%(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne object-store admin http://network:8080/v1
+-----+-----+
| Field          | Value                  |
+-----+-----+
| enabled        | True                   |
| id             | 08c18a5313f642d59de980f51666f830 |
| interface      | admin                  |
| region         | RegionOne              |
| region_id      | RegionOne              |
| service_id     | d9d7bc4b99774d3ba701e2eae93edfe2 |
| service_name   | swift                  |
| service_type   | object-store           |
| url            | http://network:8080/v1 |
+-----+-----+

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- **Swift Install (network swift-proxy)**

```
$ network> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install openstack-swift-proxy python3-memcached openssh-clients
# swift-proxy 및 관련 모듈을 설치

$ network> vi /etc/swift/proxy-server.conf
[filter:cache]
use = egg:swift#memcache
memcache_servers = controller:11211

[filter:authtoken]
paste.filter_factory = keystonemiddleware.auth_token:filter_factory


# admin_tenant_name = %SERVICE_TENANT_NAME%
# admin_user = %SERVICE_USER%
# admin_password = %SERVICE_PASSWORD%
# auth_host = 127.0.0.1
# auth_port = 35357
# auth_protocol = http
# signing_dir = /tmp/keystone-signing-swift
# 주석처리 후, 하단의 아래의 항목들을 추가

www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = swift
password = qwer1234
delay_auth_decision = true

$ network> vi /etc/swift/swift.conf
[swift-hash]
swift_hash_path_suffix = swift_shared_path
swift_hash_path_prefix = swift_shared_path
# 파일 안에 내용들을 삭제 후, 생성

$ network> swift-ring-builder /etc/swift/account.builder create 12 3
1
$ network> swift-ring-builder /etc/swift/container.builder create 12
3 1
$ network> swift-ring-builder /etc/swift/object.builder create 12 3
1

$ network> swift-ring-builder /etc/swift/account.builder add r0z0-
10.10.10.50:6202/device 100
$ network> swift-ring-builder /etc/swift/container.builder add r0z0-
10.10.10.50:6201/device 100
$ network> swift-ring-builder /etc/swift/object.builder add r0z0-
10.10.10.50:6200/device 100
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ network> swift-ring-builder /etc/swift/account.builder add r1z1-10.10.10.51:6202/device 100
$ network> swift-ring-builder /etc/swift/container.builder add r1z1-10.10.10.51:6201/device 100
$ network> swift-ring-builder /etc/swift/object.builder add r1z1-10.10.10.51:6200/device 100

$ network> swift-ring-builder /etc/swift/account.builder add r2z2-10.10.10.52:6202/device 100
$ network> swift-ring-builder /etc/swift/container.builder add r2z2-10.10.10.52:6201/device 100
$ network> swift-ring-builder /etc/swift/object.builder add r2z2-10.10.10.52:6200/device 100

$ network> swift-ring-builder /etc/swift/account.builder rebalance
$ network> swift-ring-builder /etc/swift/container.builder rebalance
$ network> swift-ring-builder /etc/swift/object.builder rebalance

$ network> chown swift. /etc/swift/*.gz
$ network> systemctl enable --now openstack-swift-proxy
$ network> firewall-cmd --add-port=8080/tcp --permanent
$ network> firewall-cmd --reload
```


- Swift Install (Storage)

```
$ storage all> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel -y install openstack-swift-account openstack-swift-container openstack-swift-object openstack-selinux xfsprogs rsync rsync-daemon openssh-clients
# swift 및 관련 모듈을 설치

$ storage all> mkfs.xfs -i size=1024 -s size=4096 /dev/sdb1
$ storage all> mkdir -p /srv/node/device
$ storage all> mount -o noatime,nodiratime /dev/sdb1 /srv/node/device
$ storage all> chown -R swift. /srv/node
# 테스트용 하드 디스크를 임포트 후, XFS 로 포맷을 진행

$ storage all> vi /etc/fstab
/dev/sdb1 /srv/node/device xfs
noatime,nodiratime 0 0
# 설정을 fstab 의 등록

$ network> scp /etc/swift/*.gz storage1:/etc/swift/
$ network> scp /etc/swift/*.gz storage2:/etc/swift/
$ network> scp /etc/swift/*.gz storage3:/etc/swift/
# 설정을 복사
```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ storage all> chown swift. /etc/swift/*.gz
$ storage all> vi /etc/swift/swift.conf
[swift-hash]
swift_hash_path_suffix = swift_shared_path
swift_hash_path_prefix = swift_shared_path

$ storage all> vi /etc/swift/account-server.conf
bind_ip = 0.0.0.0
bind_port = 6202

$ storage all> vi /etc/swift/container-server.conf
bind_ip = 0.0.0.0
bind_port = 6201

$ storage all> vi /etc/swift/object-server.conf
bind_ip = 0.0.0.0
bind_port = 6200


$ storage all> vi /etc/rsyncd.conf
pid file = /var/run/rsyncd.pid
log file = /var/log/rsyncd.log
uid = swift
gid = swift

pid file = /var/run/rsyncd.pid
log file = /var/log/rsyncd.log
uid = swift
gid = swift
address = storage1 or storage2 or storage3

[account]
path          = /srv/node
read only     = false
write only    = no
list          = yes
incoming chmod = 0644
outgoing chmod = 0644
max connections = 25
lock file =    /var/lock/account.lock

[container]
path          = /srv/node
read only     = false
write only    = no
list          = yes
incoming chmod = 0644
outgoing chmod = 0644
max connections = 25
lock file =    /var/lock/container.lock

[object]
path          = /srv/node
read only     = false
write only    = no
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
list                = yes
incoming chmod      = 0644
outgoing chmod      = 0644
max connections     = 25
lock file           = /var/lock/object.lock

[swift_server]
path                = /etc/swift
read only           = true
write only          = no
list                = yes
incoming chmod      = 0644
outgoing chmod      = 0644
max connections     = 5
lock file           = /var/lock/swift_server.lock


$ storage all> semanage fcontext -a -t swift_data_t /srv/node/device
$ storage all> restorecon /srv/node/device
$ storage all> firewall-cmd --add-
port={873/tcp,6200/tcp,6201/tcp,6202/tcp} --permanent
$ storage all> firewall-cmd --reload
# SELinux 및 방화벽을 설정

$ storage all> systemctl enable --now rsyncd \
openstack-swift-account-auditor \
openstack-swift-account-replicator \
openstack-swift-account \
openstack-swift-container-auditor \
openstack-swift-container-replicator \
openstack-swift-container-updater \
openstack-swift-container \
openstack-swift-object-auditor \
openstack-swift-object-replicator \
openstack-swift-object-updater \
openstack-swift-object
# swift 서비스를 등록 및 시작
```

- 확인

```
$ controller ~(keystone)> dnf --enablerepo=centos-openstack-
ussuri,PowerTools,epel -y install python3-openstackclient python3-
keystoneclient python3-swiftclient
# swift 사용을 위해 관련 모듈을 설치합니다.

$ controller ~(keystone)> openstack project create --domain default
--description "Swift Service Project" swiftservice
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| description | Swift Service Project                   |
| domain_id   | default                                 |
| enabled     | True                                    |
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| id          | ab658f35464e49b7a3df626e09feab91 |
| is_domain   | False                               |
| name        | swiftservice                        |
| options     | {}                                  |
| parent_id   | default                             |
| tags        | []                                  |
+-----+-----+

$ controller ~(keystone)> openstack role create SwiftOperator
+-----+-----+
| Field      | Value                               |
+-----+-----+
| description | None                               |
| domain_id  | None                               |
| id         | 3818d26e54244c1ba5d0481e9ad44e6e |
| name       | SwiftOperator                     |
| options    | {}                                  |
+-----+-----+

$ controller ~(keystone)> openstack user create --domain default --
project swiftservice --password qwer1234 swiftuser01
+-----+-----+
| Field      | Value                               |
+-----+-----+
| default_project_id | ab658f35464e49b7a3df626e09feab91 |
| domain_id        | default                             |
| enabled          | True                               |
| id              | 2ac2c69fd55a4bef95b2a8b728f131a7 |
| name            | swiftuser01                         |
| options         | {}                                  |
| password_expires_at | None                             |
+-----+-----+


$ controller ~(keystone)> openstack role add --project swiftservice
--user swiftuser01 SwiftOperator

$ controller ~(keystone)> vi ~/swift
export OS_PROJECT_DOMAIN_NAME=default
export OS_USER_DOMAIN_NAME=default
export OS_PROJECT_NAME=swiftservice
export OS_USERNAME=swiftuser01
export OS_PASSWORD=qwer1234
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export PS1='[\u@\h \W(swift)]\$ '

$ controller ~(keystone)> chmod 600 ~/swift
$ controller ~(keystone)> source ~/swift
$ controller ~(keystone)> echo "source ~/swift " >> ~/.bash_profile

$ controller ~(swift)> swift stat
Account: AUTH_ab658f35464e49b7a3df626e09feab91
Containers: 0
Objects: 0

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

Bytes: 0
Content-Type: text/plain; charset=utf-8
X-Timestamp: 1597360203.35834
X-Put-Timestamp: 1597360203.35834
Vary: Accept
X-Trans-Id: tx09982b0a02ac4b7eac244-005f35c849
X-Openstack-Request-Id: tx09982b0a02ac4b7eac244-005f35c849

$ controller ~(swift)> openstack container create test
+-----+-----+-----+-----+
| account | container | x-trans-id |
+-----+-----+-----+
| AUTH_ab658f35464e49b7a3df626e09feab91 | test |
txce00712612794927965f7-005f35c864 |
+-----+-----+-----+-----+


$ controller ~(swift)> openstack container list
+-----+
| Name |
+-----+
| test |
+-----+

$ controller ~(swift)> openstack object create testfile.txt test
$ controller ~(swift)> openstack object list test
$ controller ~(swift)> rm testfile.txt
$ controller ~(swift)> openstack object save test testfile.txt
$ controller ~(swift)> ll testfile.txt
$ controller ~(swift)> openstack object delete test testfile.txt
$ controller ~(swift)> openstack object list test

```

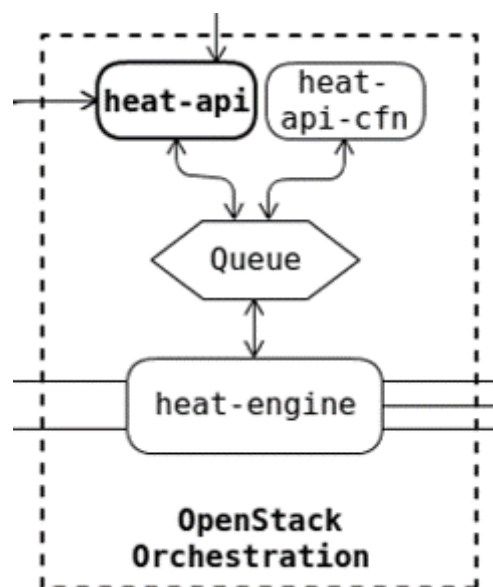
오케스트레이션 서비스 : Heat

- Heat는 템플릿과 Stack을 사용하여 자동으로 인스턴스의 리소스를 추가하거나 줄이는 서비스인 오케스트레이션을 지원하는 서비스
- 오케스트레이션은 인스턴스 생성에 대한 일련과 과정을 자동화해서 인프라를 보다 쉽게 배포할 수 있도록 하는 템플릿 기반 엔진


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- 오케스트레이션에 사용되는 템플릿 언어는 인프라, 서비스, 응용프로그램, 프로비저닝, 자동화 컴퓨팅, 스토리지, 네트워킹, 자동 스케일링 등에 사용이 가능

Heat의 구성요소와 논리 아키텍처



구성요소	역할
heat-api	RPC heat 엔진에 전송해서 요청된 API를 처리한 REST API를 제공
heat-api-cfn	AWS CloudFormation과 호환되는 AWS 타입의 Query API를 제공
heat-engine	템플릿을 생성하고, Consumer(API를 사용하려고 접근하는 애플리케이션이나 서비스)를 다시 이벤트로 제공하는 오케스트레이션의 주 작업을 수행
queue	각 서비스들이 통신을 하기 위한 서비스

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

Heat Install

- Heat User, Service, DB 생성

```
$ controller ~(keystone)> openstack user create --domain default --
project service --password qwer1234 heat
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| default_project_id | b470c69e28db47cdbfc81e06cc67f627 |
| domain_id        | default                               |
| enabled          | True                                 |
| id               | 148bafa480d84f87ba939968edb2585f |
| name             | heat                                 |
| options          | {}                                   |
| password_expires_at | None                                |
+-----+-----+


$ controller ~(keystone)> openstack role add --project service --
user heat admin

$ controller ~(keystone)> openstack role create heat_stack_owner
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| description     | None                                   |
| domain_id       | None                                   |
| id              | d46789e4326e4055aa8f6fead7c777bb |
| name            | heat_stack_owner                     |
| options         | {}                                   |
+-----+-----+

$ controller ~(keystone)> openstack role create heat_stack_user
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| description     | None                                   |
| domain_id       | None                                   |
| id              | ff45744ddbe247919034cea7c3f309e7 |
| name            | heat_stack_user                      |
| options         | {}                                   |
+-----+-----+

$ controller ~(keystone)> openstack role add --project admin --user
admin heat_stack_owner

$ controller ~(keystone)> openstack service create --name heat --
description "Openstack Orchestration" orchestration
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| description     | Openstack Orchestration               |
| enabled         | True                                  |
+-----+-----+
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| id          | 6cd5b7c7a3234b39998073587c2d9f9a |
| name        | heat                               |
| type        | orchestration                     |
+-----+-----+


$ controller ~(keystone)> openstack service create --name heat-cfn -
-description "Openstack Orchestration" cloudformation
+-----+-----+
| Field      | Value                               |
+-----+-----+
| description | Openstack Orchestration           |
| enabled     | True                               |
| id          | 2fb2087bf8da472d8c51e9fee39c93ad |
| name        | heat-cfn                           |
| type        | cloudformation                     |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne orchestration public
http://network:8004/v1/AUTH_%(tenant_id)s
+-----+-----+
| Field      | Value                               |
+-----+-----+
| enabled     | True                               |
| id          | 50481bc9998b454a9f70682132ecb026 |
| interface   | public                             |
| region      | RegionOne                           |
| region_id   | RegionOne                           |
| service_id  | 6cd5b7c7a3234b39998073587c2d9f9a |
| service_name | heat                               |
| service_type | orchestration                       |
| url         | http://network:8004/v1/AUTH_%(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne orchestration internal
http://network:8004/v1/AUTH_%(tenant_id)s
+-----+-----+
| Field      | Value                               |
+-----+-----+
| enabled     | True                               |
| id          | 1015f4c570a747349109b76b7295876c |
| interface   | internal                             |
| region      | RegionOne                           |
| region_id   | RegionOne                           |
| service_id  | 6cd5b7c7a3234b39998073587c2d9f9a |
| service_name | heat                               |
| service_type | orchestration                       |
| url         | http://network:8004/v1/AUTH_%(tenant_id)s |
+-----+-----+

$ controller ~(keystone)> openstack endpoint create --region
RegionOne orchestration admin
http://network:8004/v1/AUTH_%(tenant_id)s

```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                     |
| id          | ed21251a3f274ba6bb35061cef6cac1d      |
| interface   | admin                                   |
| region      | RegionOne                              |
| region_id   | RegionOne                              |
| service_id  | 6cd5b7c7a3234b39998073587c2d9f9a     |
| service_name | heat                                    |
| service_type | orchestration                          |
| url         | http://network:8004/v1/AUTH_%(tenant_id)s |
+-----+-----+
```

```
$ controller ~(keystone)> openstack endpoint create --region
RegionOne cloudformation public http://network:8000/v1
```


```
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                     |
| id          | fb2b67b2a13d43e1a55f775857908a5f     |
| interface   | public                                   |
| region      | RegionOne                              |
| region_id   | RegionOne                              |
| service_id  | 2fb2087bf8da472d8c51e9fee39c93ad     |
| service_name | heat-cfn                               |
| service_type | cloudformation                         |
| url         | http://network:8000/v1                 |
+-----+-----+
```

```
$ controller ~(keystone)> openstack endpoint create --region
RegionOne cloudformation internal http://network:8000/v1
```

```
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                     |
| id          | a8f4517ecf4d4370beec9e9e17183c6b     |
| interface   | internal                               |
| region      | RegionOne                              |
| region_id   | RegionOne                              |
| service_id  | 2fb2087bf8da472d8c51e9fee39c93ad     |
| service_name | heat-cfn                               |
| service_type | cloudformation                         |
| url         | http://network:8000/v1                 |
+-----+-----+
```

```
$ controller ~(keystone)> openstack endpoint create --region
RegionOne cloudformation admin http://network:8000/v1
```

```
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| enabled     | True                                     |
| id          | 66db714e538545879b7121f7150e72fc     |
| interface   | admin                                   |
+-----+-----+
```


	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

| region      | RegionOne      |
| region_id   | RegionOne      |
| service_id  | 2fb2087bf8da472d8c51e9fee39c93ad |
| service_name | heat-cfn       |
| service_type | cloudformation |
| url         | http://network:8000/v1 |
+-----+-----+

$ controller ~(keystone)> openstack domain create --description
"Stack projects and users" heat
+-----+-----+
| Field      | Value          |
+-----+-----+
| description | Stack projects and users |
| enabled     | True           |
| id          | 36fa9838b2fa43f6a6bbc95f0cdfd0a7 |
| name        | heat           |
| options     | {}             |
| tags        | []             |
+-----+-----+


$ controller ~(keystone)> openstack user create --domain heat --
password qwer1234 heat_domain_admin
+-----+-----+
| Field      | Value          |
+-----+-----+
| domain_id   | 36fa9838b2fa43f6a6bbc95f0cdfd0a7 |
| enabled     | True           |
| id          | c77bd90604254f8097aed49ea17f6fb3 |
| name        | heat_domain_admin |
| options     | {}             |
| password_expires_at | None           |
+-----+-----+

$ controller ~(keystone)> openstack role add --domain heat --user
heat_domain_admin admin

$ controller> mysql -u root -p
$ MariaDB> create database heat;
$ MariaDB> grant all privileges on heat.* to heat@'localhost'
identified by 'qwer1234';
$ MariaDB> grant all privileges on heat.* to heat@'%' identified by
'qwer1234';
$ MariaDB> flush privileges;
$ MariaDB> exit;

```

- Heat Install (Network)

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
$ Network> dnf --enablerepo=centos-openstack-ussuri,PowerTools,epel
-y install openstack-heat-api openstack-heat-api-cfn openstack-heat-
engine python3-heatclient
# Heat 및 관련 모듈을 설치
```

```
$ Network> vi /etc/heat/heat.conf
```

```
[DEFAULT]
deferred_auth_method = trusts
trusts_delegated_roles = heat_stack_owner

heat_metadata_server_url = http://network:8000
heat_waitcondition_server_url = http://network:8000/v1/waitcondition
heat_watch_server_url = http://network:8003
heat_stack_user_role = heat_stack_user

stack_user_domain_name = heat
stack_domain_admin = heat_domain_admin
stack_domain_admin_password = qwer1234
transport_url = rabbit://openstack:qwer1234@controller

[database]
connection = mysql+pymysql://heat:qwer1234@controller/heat

[clients_keystone]
auth_uri = http://controller:5000


[ec2authtoken]
auth_uri = http://controller:5000

[heat_api]
bind_host = 0.0.0.0
bind_port = 8004

[heat_api_cfn]
bind_host = 0.0.0.0
bind_port = 8000

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = heat
password = qwer1234

[trustee]
auth_plugin = password
auth_url = http://controller:5000
username = heat
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```
password = qwer1234
user_domain_name = default

$ network> chgrp heat /etc/heat/heat.conf
$ network> chmod 640 /etc/heat/heat.conf
$ network> su -s /bin/bash heat -c "heat-manage db_sync"
$ network> systemctl enable --now openstack-heat-api openstack-heat-api-cfn openstack-heat-engine
# DB를 import 시키고, heat 서비스를 등록 및 시작

$ network> firewall-cmd --add-port={8000/tcp,8004/tcp} --permanent
$ network> firewall-cmd --reload
# 방화벽을 설정
```

- 확인

```
$ controller ~(keystone)> vi sample-stack.yml
heat_template_version: 2018-08-31

description: Heat Sample Template


parameters:
  ImageID:
    type: string
    description: Image used to boot a server
  NetID:
    type: string
    description: Network ID for the server

resources:
  server1:
    type: OS::Nova::Server
    properties:
      name: "Heat_Deployed_Server"
      image: { get_param: ImageID }
      flavor: "m1.tiny"
      networks:
        - network: { get_param: NetID }

outputs:
  server1_private_ip:
    description: IP address of the server in the private network
    value: { get_attr: [ server1, first_address ] }

$ controller ~(keystone)> openstack stack create -t sample-stack.yml
--parameter "ImageID=cirros;NetID=Int_net" Sample-Stack

# controller ~(keystone)> openstack stack list
-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Stack Name | Project | Creation Time | Updated Time | Stack Status |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Stack Name | Project | Creation Time | Updated Time | Stack Status |
```

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 4cb88c32-24f9-41cf-a44d-e18593c5eb2f | Sample-Stack |
edd7025c02574d3aa2d3ab6e56208320 | CREATE_COMPLETE | 2020-08-
13T09:39:16Z | None |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+

# controller ~(keystone)> openstack server list
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | Status |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ab20d06c-955a-404b-9525-11e3e4b09484 | Heat_Deployed_Server |
ACTIVE | int_net=192.168.100.6 | cirros | m1.tiny |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```


데이터베이스 서비스 : Trove

차후 업데이트 예정입니다.

데이터 프로세싱 서비스 : Sahara

차후 업데이트 예정입니다.

베어메탈 서비스 : Ironic

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


차후 업데이트 예정입니다.

각 서비스별 트러블 슈팅

- 현재 정리집을 분실해 다시 정리 중입니다...

Keystone

- Data import시 Maria DB에 SQL문 에러가 발생할 경우
 - `/etc/keystone/keystone.conf [databases]` 하단의 DB명 User, PW 확인
 - 정상적으로 유저가 생성되었는 지 `mysql -u keystone -p$pw`로 접속하여 `show databases;` 및 `show tables`로 확인
- SQL에 연결하지 못하는 에러가 발생할 경우
 - 기본 언어셋의 부재시, MariaDB 설치시 `utf8mb4`로 설정이 필요 (MariaDB install참조)
 - 바인드 오류시에는 `/etc/mysql/mariadb.conf.d/50-server.cnf` 의 `bind address`를 `0.0.0.0`으로 교체
 - 만약 이미 테이블이 만들어졌다면 위와 같이 수정하고 다시 생성이 필요
- openstack service 생성시 internal 500
 - keystone 설치시 함께 설치했던 `httpd`가 정상적으로 실행되어 있는지를 확인
 - 오류 시 `/var/log/keystone/keystone.conf`에서 오류 이유를 확인
 - queue 오류시에는 RabbitMQ 설치시 생성했던 openstack user 및 권한을 확인하고 `keystone.conf` 파일에 [DEFAULT]에 `rabbitmq` 설정을 확인
 - [oslo...] 오류가 발견시에는 `keystone.conf`에서 [oslo...] 설정을 맞게 기입했는 지를 확인
- 정상적으로 동작하다 다른 서비스를 설치시에 인증문제

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	


- 정상적으로 동작했으나, 후에 다른 서비스 설치시 인증에러가 발생할 경우는 대부분 각 서비스의 잘못된 값을 기입한 것이 원인이나 혹은 openstack service의 endpoint를 지정할 때 실수로 동일한 포트번호 혹은 특수문자 W(ternet...W)을 (ternet..)으로 적은 것이 주 이유이며, keystone은 정상적으로 작동할 경우가 많음

Glance

- Data import시 Maria DB에 SQL문 에러가 발생할 경우
 - /etc/glance/glance.conf [databases] 하단의 DB명 User, PW 확인
 - 정상적으로 유저가 생성되었는 지 mysql -u glance -p\$pw로 접속하여 show databases; 및 show tables로 확인
- 인증관련 permission error
 - keystone 인증을 받지 않았거나, 인증파일이 오류가 있을 경우 발생
- image import시 발생하는 internal 500 server error
 - glance endpoint가 잘못될 경우 발생, openstack endpoint list로 확인 후 수정

Nova


- Nova 또한 다수의 기술이 사용되고 오류가 많아, 그 연관관계가 매우 복잡한 것으로 알고 있음. (트러블 슈팅이지만 원인관계가 복잡해 정확도가 부족하다는 사실 인지)
- Data import시 Maria DB에 SQL문 에러가 발생할 경우
 - /etc/nova/nova.conf [databases] 하단의 DB명 User, PW 확인

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- 정상적으로 유저가 생성되었는 지 `mysql -u nova -p$pw`로 접속하여 `show databases;` 및 `show tables`로 확인
- nova의 경우 데이터베이스가 1개가 아닌 다수개인 것을 반드시 주의할 것!
- 정상적으로 설치가 완료되었음에도 compute가 openstack compute service list에 출력되지 않을 때는 controller node에서 compute node를 찾기위한 검색어를 입력해야 함(설치참조)
- 인스턴스 생성시 network 없이는 생성이 되는 데, network 포함시 에러가 출력될 경우
 - 위와 같은 에러 발생시 compute node에 `/var/log/neutron/openvswitch.log`를 확인
 - compute node 또한 neutron의 가상화 서비스를 설치하며, 이에 따라 nova 안에 neutron 서비스에 대한 설정값을 기입되는 것을 확인해야 하며, 기본적인 (neutron ml2, openvswitch...)에 대한 설정 값을 확인
- Import 된 Image파일을 읽어오지 못할 경우
 - Glance service가 실행되어 있는 지를 확인
 - `/etc/nova/nova.conf`에서 [glance] 설정 값이 맞게 설정되어 있는 지를 확인
- 인스턴스 생성시 발생하는 internal 500 server error
 - `/var/log/nova/nova-conductor.log` 확인
 - nova endpoint가 잘못될 경우 발생, openstack endpoint list로 확인 후 수정
 - NTP server 확인


Neutron

- Neutron 또한 다수의 기술이 사용되고 오류가 많아, 그 연관관계가 매우 복잡한 것으로

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

알고 있음. (트러블 슈팅이지만 원인관계가 복잡해 정확도가 부족하다는 사실 인지)


- 여기서는 Linuxbridge가 아닌 openvswtch로 기준이 잡혀져 있음
- Data import시 Maria DB에 SQL문 에러가 발생할 경우
 - /etc/neutron/neutron.conf [databases] 하단의 DB명 User, PW 확인
 - 정상적으로 유저가 생성되었는 지 mysql -u neutron -p\$pw로 접속하여 show databases; 및 show tables로 확인
- 네트워크 인터페이스가 작동하지 않을 때, 혹은 재시작 후 작동하지 않을 때
 - neutron은 네트워크 가상화 인터페이스를 생성 후, 원래의 네트워크 인터페이스를 여기에 스위칭 하는 방식으로 사용
 - SDN 기술이 활용되며 이에 따라 오류가 발생하면 직접 수정하는 방식이 필요
- 특정 network agent가 동작을 하지 않을 경우
 - openstack network agent list를 입력시 특정 서비스가 동작을 하지 않을 경우, 특정 노드에서 설정파일들의 확인이 필요 ex) /etc/neutron/plugins/ml2/...
 - 서비스들이 실행되어 있는 지, 혹은 수정 후 재시작 했는 지 또한 확인
- 네트워크 생성시 발생하는 internal 500 server error
 - /var/log/neutron/neutron의 로그들을 확인
 - 네트워크의 포트와 가상화된 포트가 스위칭 되지 않았을 때도 발생
 - neutron endpoint가 잘못될 경우 발생, openstack endpoint list로 확인 후 수정
 - NTP server 확인
- Router 생성시 내부대역 혹은 외부대역이 작동을 하지 않을 때

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- 내부대역이 동작하지 않을 때는 compute node(nova)의 openvswitch를 확인
- 외부대역이 동작하지 않을 때는 network node(neutron)의 openvswitch를 확인

Cinder

- Data import시 Maria DB에 SQL문 에러가 발생할 경우
 - /etc/cinder/cinder.conf [databases] 하단의 DB명 User, PW 확인
 - 정상적으로 유저가 생성되었는 지 mysql -u cinder -p\$pw로 접속하여 show databases; 및 show tables로 확인
- volume 생성시의 오류 not found
 - volume 생성시의 오류는 Cinder 생성시 생성한 VG에 대한 오류로 이에 대한 확인이 필요
 - vgdisplay 명령어를 입력하여 확인
 - /etc/fstab에 등록되어 있는 지를 확인
- volume 생성시의 오류 internal 500 server error
 - /var/log/cinder/cinder*의 로그들을 확인
 - cinder endpoint가 잘못될 경우 발생, openstack endpoint list로 확인 후 수정
 - NTP server 확인
- 인스턴스 생성시의 volume을 사용하면 에러가 발생하는 경우
 - volume 생성시 어느한도의 시간이 경과하면 작업이 취소되는 설정이 있어 이에 대한 해제가 필요
 - Cinder 서비스와 Nova 서비스가 서로 간의 통신이 불가능한 문제로 Nova의 설정파

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

일과 Cinder의 설정파일의 설정 값들을 확인


- Volume 생성시 부팅 볼륨 (image)가 생성되지 않는 경우
 - glance의 대한 설정 값 오류로 발생 cinder.conf에 [glance] 설정 값 확인

Swift

- Swift는 독자적으로 생성되는 유일한 서비스로 다른 서비스의 영향을 주는 경우도 많음.
(트러블 슈팅이지만 원인관계가 복잡해 정확도가 부족하다는 사실 인지)
- container 생성시의 internal 500 server error
 - swift endpoint 생성시 발생할 수 있으며, openstack endpoint list로 확인 및 수정
- container 생성시의 permission error
 - storage의 설정 값들의 대한 오류로 발생할 수 있음
 - storage의 설정파일 및 서비스 실행여부 확인

Heat

- Data import시 Maria DB에 SQL문 에러가 발생할 경우
 - /etc/heat/heat.conf [databases] 하단의 DB명 User, PW 확인
 - 정상적으로 유저가 생성되었는 지 mysql -u heat -p\$pw로 접속하여 show databases; 및 show tables로 확인

	Report	Version	Last Modified	OpenStack 아키텍처 분석
	Final Report	3.0	2020.11.14	

- heat 서비스 사용시 발생하는 internal 500 server error
 - heat endpoint 설정이 잘못될 경우 발생, openstack endpoint list로 확인 후 수정
 - heat 설정 값 /etc/heat/heat.conf에 설정 값이 잘못되었을 경우 발생 확인 후 수정

Horizon

- Horizon 서비스가 접속이 불가능할 경우
 - /var/log/httpd/*.log 파일들을 확인
 - httpd service 상태확인
- 내부에서는 접속가능하나 외부에서 접속이 불가능할 경우
 - httpd의 접속허용 확인 /etc/httpd/httpd.conf VirtualHost, Directory, File 확인