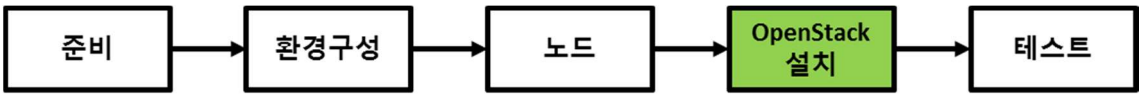
4. OpenStack 설치 및 구성



이 절에서는 컨트롤러 노드와 컴퓨트 노드에 OpenStack 을 구성하는 서비스 들을 실제로 설치하는 과정을 설명한다. 이 절에서의 진행 순서는 다음과 같다.

- 공통 서비스 설치
- 자격증명(Identity) 서비스 - Keystone
- 이미지(Image) 서비스 - Glance
- 컴퓨트 서비스 - Nova
- 네트워킹 서비스 - Neutron
- 대시보드 서비스 - Horizon
- 블록스토리지 서비스 - Cinder

이 절에서 설명할 내용은 OpenStack 의 설지 가이드를 기본으로 진행한다.

4.1 공통 서비스 설치

먼저 OpenStack 서비스에서 공통으로 사용될 패키지와 서비스 들을 설치한다. 다음의 모든 과정은 OS 에 루트로 로그인하여 실행되어야 한다.

4.1.1 설치에 필요한 암호 및 보안

OpenStack 에서 설치되는 다양한 서비스 들에는 암호가 필요하다. 서비스 별 설치 과정에서 필요한 암호를 미리 다음의 표에 정리해 둔다. 아래의 표에 이 글에서 사용할 자격증명(Keystone), 이미지(Glance), 블록 스토리지(Cinder), 네트워킹(Neutron), 컴퓨트(Nova), 대시보드(Horizon) 서비스 및 공통 서비스에 관련된 암호가 정리되어 있고
(이 글에서 입력 시에는 *italic* 체로 표시한다. 실제 설치 시에는 원하는 암호로 변경하기 바란다.)

Table 9 설치에 필요한 암호

암호 이름	값	설명
DB(데이터베이스) 암호	dbpassword	데이터베이스 루트 암호
ADMIN_PASS	gotocloud_admin	OpenStack admin 암호
DEMO_PASS	gotocloud_demo	OpensStack demo 암호

RABBIT_PASS	rabbitpass	RabbitMQ 사용자 암호
KEYSTONE_DBPASS	keystonedbpass	자격증명(Keystone) 서비스 DB 암호
GLANCE_DBPASS	glancedbpass	이미지(Glance) 서비스 DB 암호
GLANCE_PASS	glancepass	이미지(Glance) 서비스 glance 암호
CINDER_DBPASS	cinderdbpass	블록 스토리지(Cinder) 서비스 DB 암호
CINDER_PASS	cinderpass	블록 스토리지(Cinder) 서비스 cinder 암호
NEUTRON_DBPASS	neutrondbpass	네트워킹(Neutron) 서비스 DB 암호
NEUTRON_PASS	neutronpass	네트워킹(Neutron) 서비스 neutron 암호
NOVA_DBPASS	novadbpass	컴퓨트(Nova) 서비스 DB 암호
NOVA_PASS	novapass	컴퓨트(Nova) 서비스 nova 암호
DASH_DBPASS	horizondbpass	대시보드(Horizon) 서비스 DB 암호

이와 함께, 각 서비스 별로 방화벽 설정이 필요할 수 있는데, 설치 진행 과정이나 완료 후에 방화벽 설정에 관해서는 별도로 설명할 예정이므로, 이 절에서는 운영체제에서 기본적으로 제공하고 있는 설정 그대로 진행하도록 한다.

4.1.2 NTP(Network Time Protocol) 설치

일반적으로 실제 운영 환경에서는 Figure 10과 같이 컨트롤러 노드가 직접 인터넷 상의 시간동기화 서버와 통신하도록 하기 보다는 조직내에 설치되어 있는 별도의 시간 동기화 서버와 통신하도록 설정하여 컨트롤러 노드가 인터넷에 직접 연결되지 않도록 구성한 후에, 컴퓨트 노드등 OpenStack 노드들이 컨트롤러와 관리 네트워크를 통해 시간을 동기화하도록 설정하는 것이 보편적이다.

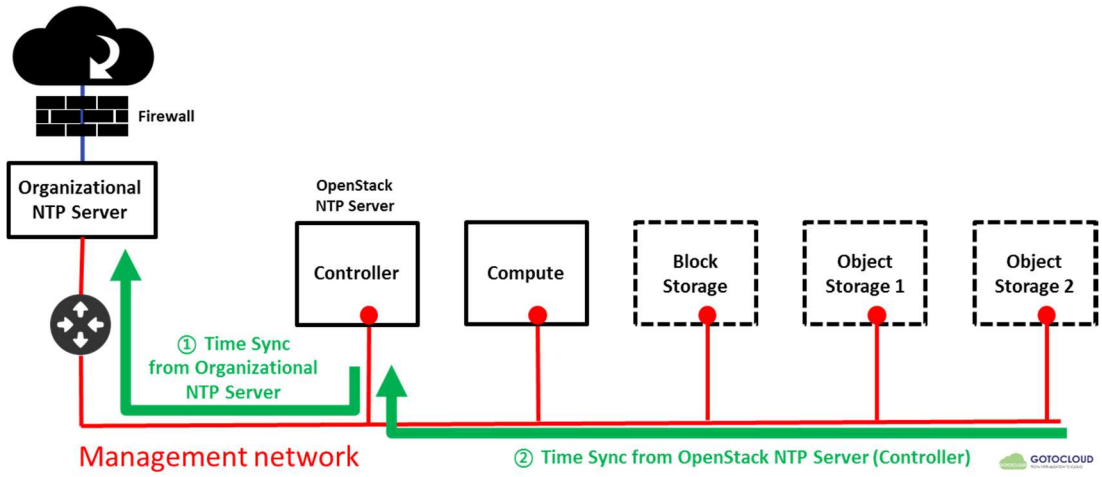


Figure 10 실제 운영 환경에서의 시간 동기화 서버 구성

이 글에서는 컨트롤러 노드는 외부의 시간 동기화 서버와 동기화하고, 컴퓨트 노드등 다른 노드는 컨트롤러를 시간 동기화 서버로 사용하여, 관리 네트워크를 통해 모든 노드간의 시간을 동기화하도록 구성한다. Figure 11 에서 볼 수 있듯이, 컨트롤러 노드는 Install Network 를 통해 외부 인터넷 상의 시간 동기화 서버와 동기화하고, 컴퓨트 노드를 비롯한 OpenStack 노드들은 컨트롤러 노드를 시간 동기화 서버로 설정하여 시간을 동기화 하게 된다.

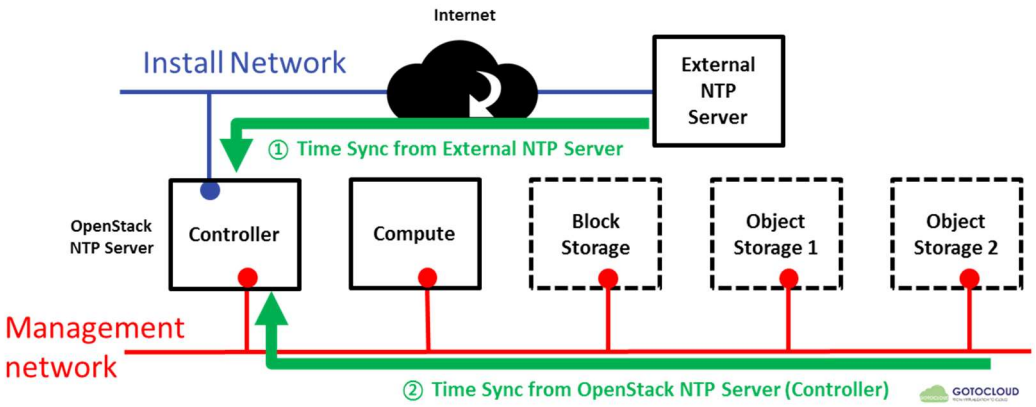


Figure 11 OpenStack 노드 시간 동기화 구성

1. 컨트롤러 노드 시간 동기화 설정

a. 컨트롤러 노드에서 시간 동기화를 위한 chrony 패키지를 설치한다.

```
# yum -y install chrony
```

b. /etc/chrony.conf 에 외부 인터넷의 시간 동기화 서버의 주소를 추가한다. 이 글에서는 CentOS 에서 제공하는 시간 동기화 서버를 사용한다.

```
# vi /etc/chrony.conf
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
server 2.centos.pool.ntp.org iburst
server 3.centos.pool.ntp.org iburst
```

c. 컴퓨트 노드 등이 시간 동기화 서버인 컨트롤러 노드로 연결할 수 있도록 관리 네트워크 대역인 10.0.0.0/24 를 허용한다.

```
# vi /etc/chrony.conf
allow 10.0.0.0/24
```

d. chronyd 서비스가 자동으로 실행되도록 서비스를 등록하고, 실행한다.

```
# systemctl enable chronyd.service
# systemctl start chronyd.service
```

2. 컴퓨트 노드 시간 동기화 설정

a. 컴퓨트 노드와 다른 노드에서 시간 동기화를 위한 chrony 패키지를 설치한다.

```
# yum -y install chrony
```

b. /etc/chrony.conf 에 컨트롤러 노드를 시간 동기화 서버로 등록한다.

```
# vi /etc/chrony.conf
server controller iburst
```

c. chronyd 서비스가 자동으로 실행되도록 서비스를 등록하고, 실행한다.

```
# systemctl enable chronyd.service
# systemctl start chronyd.service
```

3. 시간 동기화 서버 설정 확인

a. 컨트롤러 노드에서 시간 동기화가 정상적으로 동작하는지를 다음의 명령으로 확인한다. 접속 가능한 시간동기화 서버가 표시되며, * 표시가 있는 서버가 현재 동기화에 사용되는 서버이다.

```
# chronyc sources
210 Number of sources = 4
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^- mail.funix.net              3   7   377    60  -1955us[-1955us] +/-  138ms
^* 114.207.245.166             2   6   377   126   +275us[ +358us] +/-   42ms
^- 106.247.248.106             3   7   377    60    -35ms[  -35ms] +/-  325ms
^- 115.90.134.38               2   7   377    59   +682us[ +682us] +/-  412ms
```

b. 컴퓨트 노드에서 시간 동기화가 정상적으로 동작하는지를 다음의 명령으로 확인한다. 컨트롤러 노드가 시간 동기화 서버로 동작하고 있는 것을 확인할 수 있다.

```
# chronyc sources
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^* controller                  3   6   177     3    +86us[+1481us] +/-   37ms
```

4.1.3 OpenStack 패키지 설치

1. CentOS에서는 OpenStack 패키지 저장소를 활성화하는 RPM이 extra 저장소에서 제공되므로, 모든 노드에서 다음의 명령어를 실행하여 OpenStack 저장소를 활성화한다.

```
# yum -y install centos-release-openstack-mitaka
```

2. 모든 노드에서 패키지 업그레이드를 수행한다.

```
# yum -y upgrade
```

3. OpenStack 클라이언트를 설치한다.

```
# yum -y install python-openstackclient
```

4. CentOS 에서는 보안정책 관리를 위해 SELinux 가 enforcing 모드로 기본으로 활성화되어 있는데 이를 permissive 모드로 변경하고, 시스템 시작시에 적용되도록 /etc/sysconfig/selinux 파일을 수정한다.(enforcing 모드일 경우 추가 디스크를 마운트 할 때, 쓰기가 허용되지 않는다.)

```
# setenforce 0
# vi /etc/sysconfig/selinux
...
SELINUX=permissive
...
```

5. 설치 OpenStack 서비스에 대한 보안정책 관리를 위하여 openstack-selinux 패키지를 설치한다.

```
# yum –y install openstack-selinux
```

4.1.4 SQL 서버 설치 및 구성

대부분의 OpenStack 서비스 들이 사용하는 정보는 SQL 데이터베이스를 사용하여 저장하게 된다. 일반적인 구성에서는 컨트롤러 노드에서 SQL 데이터베이스가 실행되는데, 대규모의 실제 운영 구성에서는 별도의 SQL 서버를 분리하여 구성할 수도 있다. 이 글에서는 CentOS에서 기본적으로 제공하는 MariaDB를 컨트롤러 노드에 설치하여 사용한다.

1. 컨트롤러 노드에 MariaDB 패키지를 설치한다.

```
# yum –y install mariadb mariadb-server python2-PyMySQL
```

2. /etc/my.cnf.d/openstack.cnf 파일을 생성하여 다음과 같이 편집한다.

a. [mysqld] 섹션에 다른 노드에서 컨트롤러 노드에 설치되어 있는 데이터베이스에 관리 네트워크를 통해 접속할 수 있도록 bind-address 키를 컨트롤러 노드의 IP 주소인 10.0.0.11로 설정한다.

```
# vi /etc/my.cnf.d/openstack.cnf
[mysqld]
bind-address = 10.0.0.11
```

b. [mysqld] 섹션에 UTF-8 문자 세트 활성화 및 기타의 설정을 다음과 같이 추가한다.

```
# vi /etc/my.cnf.d/openstack.cnf
[mysqld]
default-storage-engine = innodb
innodb_file_per_table
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

3. 데이터베이스 서비스가 시스템 시작시에 자동으로 실행되도록 등록한다.

```
# systemctl enable mariadb.service
# systemctl start mariadb.service
```

4. mysql_secure_installation 스크립트를 실행하여 데이터베이스 서비스 보안을 강화한다. 이 때 데이터베이스 root 암호를 Table 9 에서 설정한 *dbpassword*로 입력한다.

```
# mysql_secure_installation
...
Enter current password for root(enter for none):
...
Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
```

4.1.5 메시지 큐 설치 및 구성

OpenStack에서는 서비스가 간의 작업과 상태 정보 교환 및 조정에 메시지 큐(message queue)를 사용한다. 메시지 큐 서비스는 일반적으로 컨트롤러 노드에서 동작하는데 OpenStack은 RabbitMQ, Qpid, ZeroMQ 등 다양한 메시지 큐 서비스들을 지원하고 있다. 이 글에서는 CentOS에서 지원하는 RabbitMQ 메시지 큐 서비스를 사용한다.

1. 컨트롤러 노드에 RabbitMQ 패키지를 설치한다.

```
# yum -y install rabbitmq-server
```

2. RabbitMQ 메시지 큐 서비스가 시스템 시작시에 자동으로 실행되도록 등록한다.

```
# systemctl enable rabbitmq-server.service
# systemctl start rabbitmq-server.service
```

3. RabbitMQ 메시지 큐 사용자에 openstack 사용자를 추가한다. RabbitMQ 사용자 암호로 설정한 *rabbitpass* 를 사용한다.

```
# rabbitmqctl add_user openstack rabbitpass
Creating user "openstack" ...
```

4. openstack 사용자에 대해 구성, 쓰기 및 읽기 권한을 부여한다.

```
# rabbitmqctl set_permissions openstack ".*" ".*" ".*"
Setting permissions for user "openstack" in vhost "/" ...
```

4.1.6 Memcached 설치 및 구성

OpenStack 자격증명(Identity) 서비스에서는 memcached 서비스를 사용하여 토큰을 캐싱하는데, 일반적으로 memcached 서비스는 컨트롤러에서 실행된다.

1. 컨트롤러 노드에 Memcached 패키지를 설치한다.

```
# yum -y install memcached python-memcached
```

2. Memcached 서비스가 시스템 시작시에 자동으로 실행되도록 등록한다.

```
# systemctl enable memcached.service
# systemctl start memcached.service
```