

개방형 PaaS 플랫폼 고도화 및 개발자 지원환경 개발

설치가이드

Monasca Server(API)

|  |  |  |
| --- | --- | --- |
| 작성자 | 신충섭 (인) | 2017.9.25 |
| 검토자 | 조인호 (인) | 2017.9.26 |
| 승인자 | 정기봉 (인) | 2017.9.27 |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

개 정 이 력

| 버전 | 작성일 | 변경내용[[1]](#footnote-1) | 작성자 | 승인자 |
| --- | --- | --- | --- | --- |
| 1.0 | 2017.09.25 | 최초작성 | 조인호 | 정기봉 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목차

[1. 개요 4](#_Toc490058880)

[1.1. 문서 목적 4](#_Toc490058881)

[1.2. 범위 4](#_Toc490058882)

[1.3. 참고 자료 4](#_Toc490058883)

[2. Pre-Requisite(전제조건) 5](#_Toc490058884)

[4. MariaDB 설치 및 데이터베이스 설정 6](#_Toc490058885)

[5. Apache Zookeeper 설치 8](#_Toc490058886)

[6. Apache Kafka 설치 9](#_Toc490058887)

[7. Apache Storm 설치 12](#_Toc490058888)

[8. InfluxDB 설치 15](#_Toc490058889)

[9. Monasca Persister 설치 17](#_Toc490058890)

[10. Monasca Comon 설치 20](#_Toc490058891)

[11. Monasca Thresh 설치 20](#_Toc490058892)

[12. Monasca Notification 설치 26](#_Toc490058893)

[13. Monasca API 설치 29](#_Toc490058894)

[14. Elasticsearch 관련 프로그램 설치 38](#_Toc490058895)

[14.1. Elasticserarch 서버 설치 38](#_Toc490058896)

[14.2. logstash 설치 40](#_Toc490058897)

[ Reference : Cross-Project(Tenant) 사용자 추가 및 권한 부여 43](#_Toc490058898)

1. 개요
   1. 문서 목적

본 문서(설치가이드)는, IaaS(Infrastructure as a Service) 중 하나인 Openstack 기반의 Cloud 서비스 상태 및 자원 정보, 그리고 VM Instance의 시스템 정보를 수집 및 관리하고, 사전에 정의한 Alarm 규칙에 따라 실시간으로 모니터링하여 관리자에게 관련 정보를 제공하기 위한 서버를 설치하는데 그 목적이 있다.

* 1. 범위

본 문서의 범위는 Openstack 모니터링을 위한 오픈소스인 Monasca 제품군의 설치 및 관련

S/W(Kafka, Storm, Zookeeper, InfluxDB, MariaDB) 설치하기 위한 내용으로 한정되어 있다.

* 1. 참고 자료

<https://wiki.openstack.org/wiki/Monasca>

<http://kafka.apache.org/quickstart> **(version: 2.9.2)**

<http://storm.apache.org/releases/current/Setting-up-a-Storm-cluster.html> **(version 1.0.0)**

<https://zookeeper.apache.org/doc/r3.3.4/zookeeperStarted.html>

<https://docs.influxdata.com/influxdb/v1.2/introduction/installation/>

<https://mariadb.org/mariadb-10-2-7-now-available/>

1. Pre-Requisite(전제조건)

* Monasca Server를 설치하기 위해서는 Bare Metal 서버 또는 Openstack 에서 생성한 Instance(Ubuntu 기준, Flavor - x1.large 이상)가 준비되어 있어야 한다.
* Openstack Cross-tenant 설정이 되어 있어야 한다.

**\*\* Reference : Cross-Project(Tenant) 사용자 추가 및 권한 부여 (openstack 기준)**

* Monasca Server 설치에 필요한 프로그램 리스트 및 버전은 아래 사항을 참조한다..
* Monasca Server 를 설치하기에 필요한 프로그램을 사전에 설치한다.
* **설치 환경은 Ubuntu 14.04 기준으로 작성**하였다.

\*\* 설치 프로그램 리스트 및 버전 참조 [순서]

ᆞ MariaDB (10.2.x) (<https://mariadb.org/>) : Alarm 설정 및 관련 정보 관리

ᆞ Apache Zookeeper (3.3.2) (<https://zookeeper.apache.org/>) : 분산 코디네이터

ᆞ Apache Kafka (2.9.2) (<https://kafka.apache.org/>) : 메세지 큐 시스템

ᆞ Apache Storm (1.0.0) (<http://storm.apache.org/>) : 실시간 데이터 스트리밍 처리

ᆞ InfluxDB (1.2.x) (<https://www.influxdata.com/>) : 시스템 메트릭스 정보 관리

ᆞ ElasticSearch (5.x) (<https://www.elastic.co/kr/>) : 시스템 로그 정보 관리

ᆞ Monasca Persister (1.6.0) (<https://github.com/openstack/monasca-persister>)

: Monasca API를 통해 전달된 시스템 메트릭스 정보를 influxDB에 저장/관리

ᆞ Monasca Thresh (1.4.0) (<https://github.com/openstack/monasca-thresh>)

: Monasca API를 통해 전달된 시스템 메트릭스를 실시간 분석하여 Alarm 처리

ᆞ Monasca Notification (1.6.0) (<https://github.com/openstack/monasca-notification>)

: Monasca Thresh 를 통해 발생된 Alarm 정보를 관리자에게 전송

ᆞ Monasca API (2.0.0) (<https://github.com/openstack/monasca-api>)

: Monasca Agent를 통해 수집된 시스템 메트릭스 정보를 전송받아 처리하는 API 서버

\*\* 설치 전 사전에 설치되어 있어야 하는 프로그램

1. install git

sudo apt-get update

sudo apt-get install -y git

1. install jdk & python

sudo add-apt-repository ppa:openjdk-r/ppa

sudo apt-get update

sudo apt-get install openjdk-8-jdk python-pip python-dev

sudo apt-get install python-keystoneclient

1. install Maven

sudo apt-get install maven

1. MariaDB 설치 및 데이터베이스 설정
2. MariaDB public key 가져오기

|  |
| --- |
| **$ sudo apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80 0xcbcb082a1bb943db** |

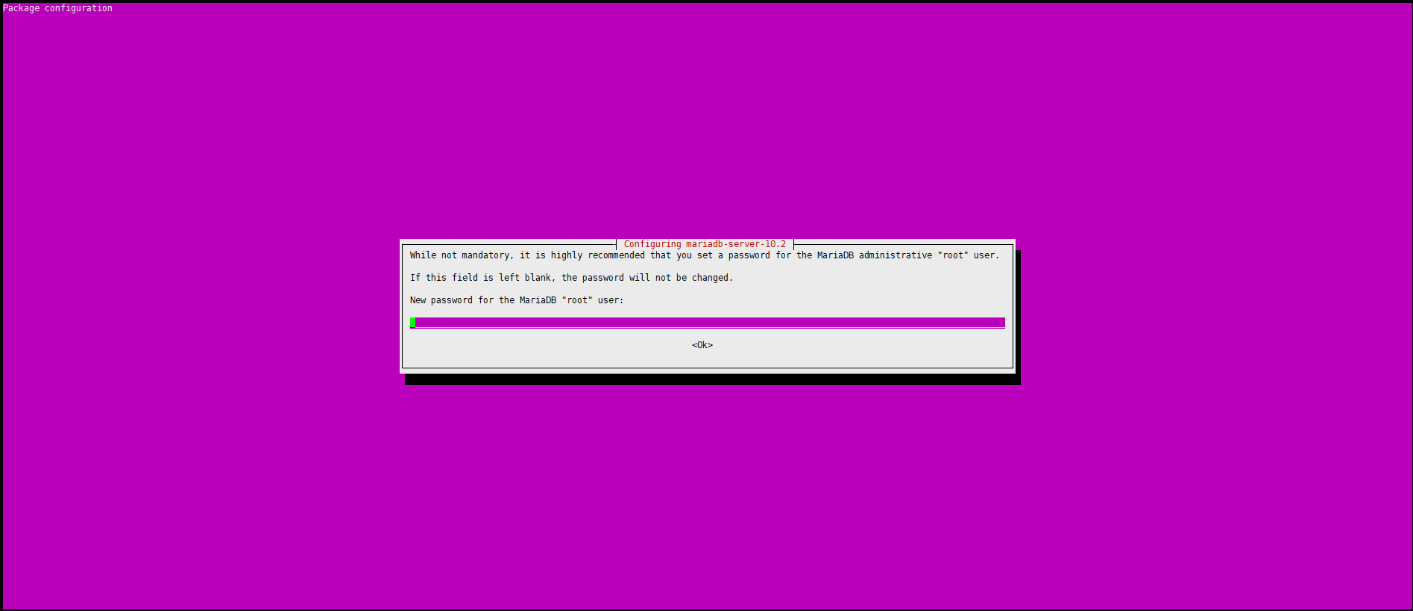
1. MariaDB repository 정보 등록.

|  |
| --- |
| **$ sudo vi /etc/apt/sources.list.d/mariadb.list**  deb [arch=amd64,i386] http://mirror.jmu.edu/pub/mariadb/repo/10.2/ubuntu trusty main  deb-src http://mirror.jmu.edu/pub/mariadb/repo/10.2/ubuntu trusty main |

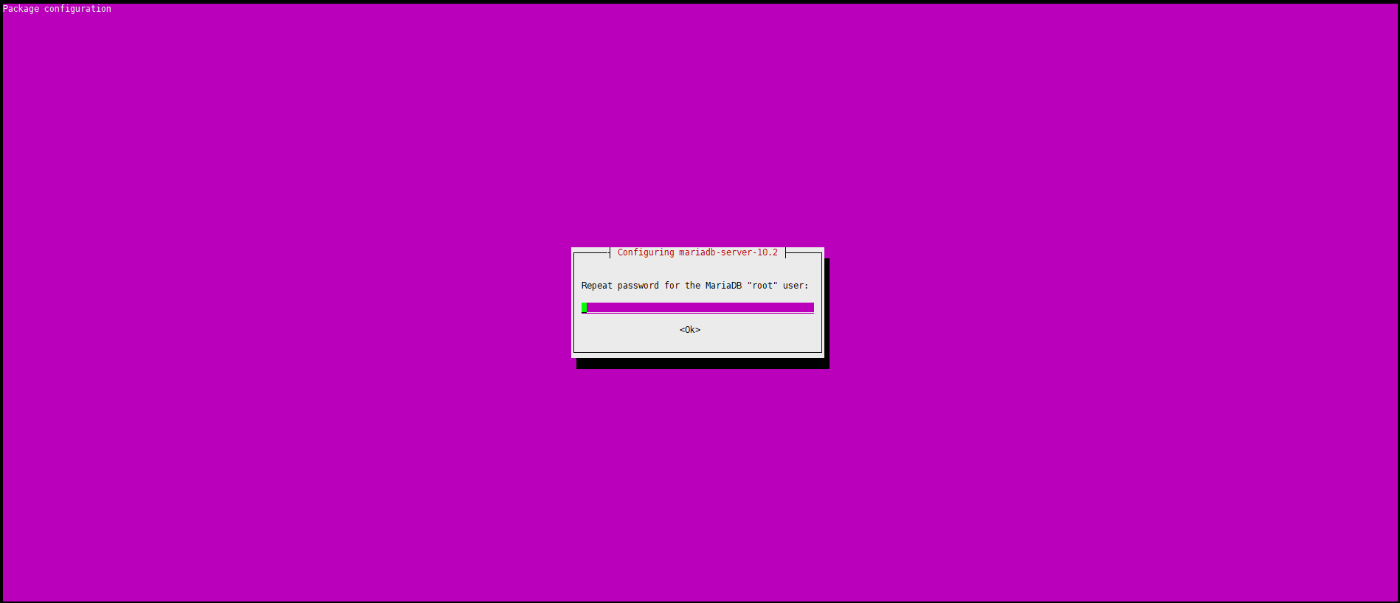
1. MariaDB 설치

|  |
| --- |
| **$ sudo apt-get update**  **$ sudo apt-get install mariadb-server** |

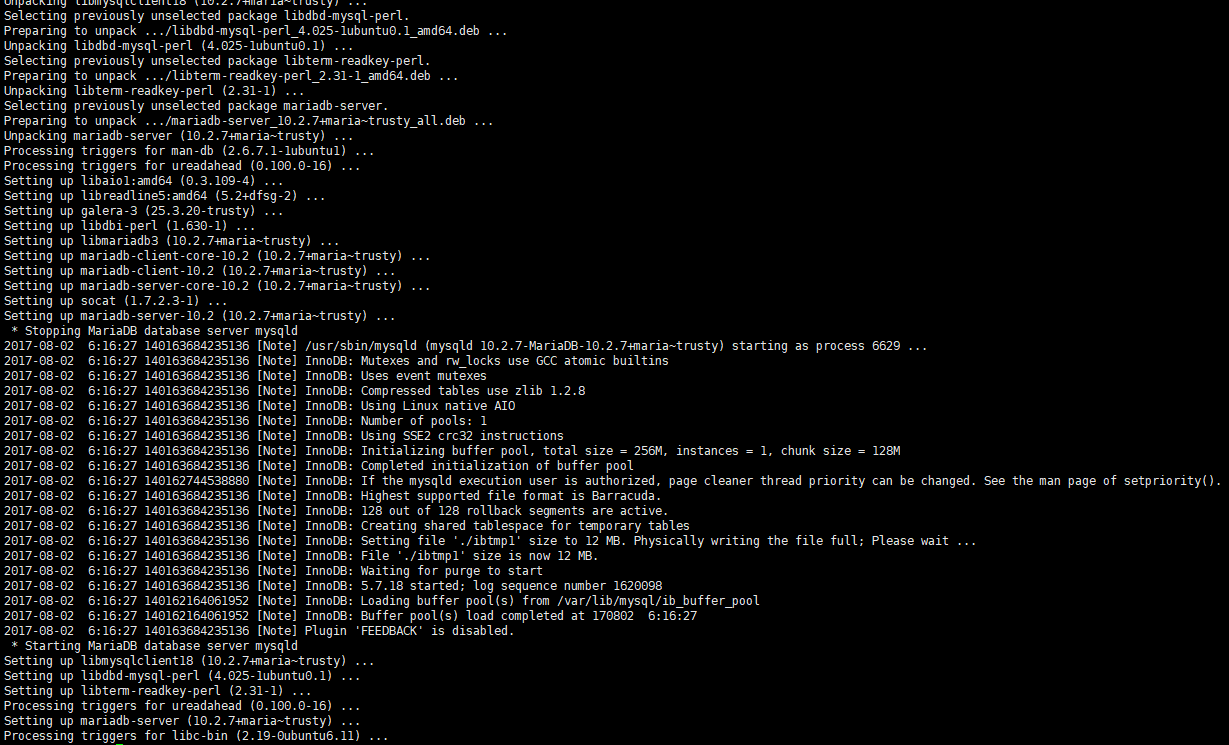
1. MariaDB root 계정의 패스워드 입력



1. MariaDB root 계정의 패스워드 확인



1. MariaDB 설치 완료 확인



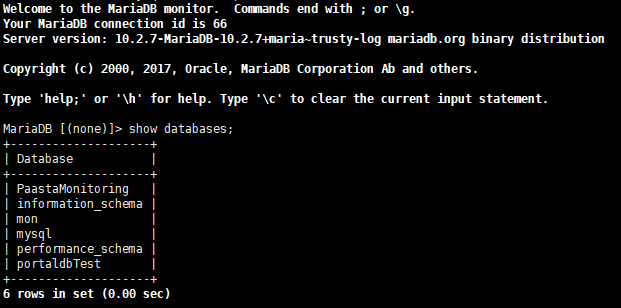
|  |
| --- |
| **$ mysql –u root –p”패스워드”** |

1. Monasca Server 관련 데이터베이스 다운로드 및 등록

|  |
| --- |
| **$ sudo apt-get install unzip**  **$ wget --no-check-certificate** <https://www.shaunos.com/wp-content/uploads/2016/09/mon_mysql.zip>  **$ unzip mon\_mysql.zip**  **# mon\_mysql.sql 파일의 monasca 사용자의 패스워드를 변경한다.**  **# Line 234,235**  **$ mysql –u root –p”패스워드” < mon\_mysql.sql** |

1. Monasa Database 확인

|  |
| --- |
| **$ mysql –u root –p”패스워드”** |



* **“mon” 데이터베이스의 존재 여부를 확인한다.**

1. Apache Zookeeper 설치
2. Apache zookeeper 다운로드 및 디렉토리 이동

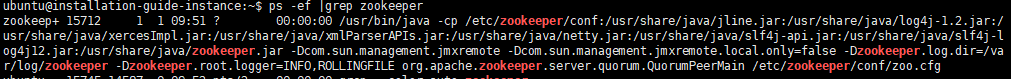
|  |
| --- |
| **$ sudo apt-get install -y zookeeper zookeeperd zookeeper-bin** |

1. Apache zookeeper 사용자 생성

|  |
| --- |
| **$ sudo useradd zookeeper -U -r** |

1. 확인

|  |
| --- |
| **$ ps -ef |grep zookeeper** |



1. Apache Kafka 설치
2. Apache kafka 다운로드

|  |
| --- |
| **$ wget http://apache.mirrors.tds.net/kafka/0.8.1.1/kafka\_2.9.2-0.8.1.1.tgz** |

1. 압축해제 및 서비스 디렉토리 변경 (Optional)

|  |
| --- |
| **$ tar zxf kafka\_2.9.2-0.8.1.1.tgz**  **$ mv kafka\_2.9.2-0.8.1.1 kafka**  **$ sudo mv kafka /opt/** |

1. 서비스 링크 생성

|  |
| --- |
| **$ sudo ln -s /opt/kafka/config /etc/kafka** |

1. Apache kafka 서비스 시작 스크립트 생성

|  |
| --- |
| **$ sudo vi /etc/init/kafka.conf**  ---  description "Kafka"  start on runlevel [2345]  stop on runlevel [!2345]  respawn  limit nofile 32768 32768  # If zookeeper is running on this box also give it time to start up properly  pre-start script  if [ -e /etc/init.d/zookeeper ]; then  /etc/init.d/zookeeper restart  fi  end script  # Rather than using setuid/setgid sudo is used because the pre-start task must run as root  exec sudo -Hu kafka -g kafka KAFKA\_HEAP\_OPTS="-Xmx1G -Xms1G" JMX\_PORT=9997 /opt/kafka/bin/kafka-server-start.sh /etc/kafka/server.properties |

1. kafka 서비스 설정

|  |
| --- |
| **$ vi /etc/kafka/server.properties**  ---  # Hostname the broker will bind to. If not set, the server will bind to all interfaces  # hostname 정보 설정  host.name=**localhost**  ...  # Hostname the broker will advertise to producers and consumers. If not set, it uses the  # value for "host.name" if configured. Otherwise, it will use the value returned from  # java.net.InetAddress.getCanonicalHostName().  # hostname 정보 설정  advertised.host.name=**localhost**  ...  # A comma seperated list of directories under which to store log files  # 로그 파일을 저장할 디렉토리 설정  log.dirs=**/opt/kafka/logs**  ... |

1. apache kafka 사용자 및 필요한 디렉토리 설정

|  |
| --- |
| **$ sudo useradd kafka -U -r**  **$ sudo mkdir /var/kafka**  **$ sudo mkdir /opt/kafka/logs**  **$ sudo chown -R kafka. /var/kafka/**  **$ sudo chown -R kafka. /opt/kafka/logs** |

1. apache kafka 서비스 시작

|  |
| --- |
| **$ sudo service kafka start** |

1. 확인

|  |
| --- |
| **$ sudo tail -10f /var/log/upstart/kafka.log**  --- 아래와 같이 정상적인 로그가 보인다면 성공 ---  [2017-08-07 06:22:29,676] INFO [Kafka Server 0], starting (kafka.server.KafkaServer)  [2017-08-07 06:22:29,678] INFO [Kafka Server 0], Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)  [2017-08-07 06:22:29,861] INFO Found clean shutdown file. Skipping recovery for all logs in data directory '/opt/kafka/logs' (kafka.log.LogManager)  [2017-08-07 06:22:29,863] INFO Starting log cleanup with a period of 60000 ms. (kafka.log.LogManager)  [2017-08-07 06:22:29,868] INFO Starting log flusher with a default period of 9223372036854775807 ms. (kafka.log.LogManager)  [2017-08-07 06:22:29,914] INFO Awaiting socket connections on localhost:9092. (kafka.network.Acceptor)  [2017-08-07 06:22:29,916] INFO [Socket Server on Broker 0], Started (kafka.network.SocketServer)  [2017-08-07 06:22:30,023] INFO Will not load MX4J, mx4j-tools.jar is not in the classpath (kafka.utils.Mx4jLoader$)  [2017-08-07 06:22:30,092] INFO 0 successfully elected as leader (kafka.server.ZookeeperLeaderElector)  [2017-08-07 06:22:30,220] INFO Registered broker 0 at path /brokers/ids/0 with address localhost:9092. (kafka.utils.ZkUtils$)  [2017-08-07 06:22:30,239] INFO [Kafka Server 0], started (kafka.server.KafkaServer)  [2017-08-07 06:22:30,328] INFO New leader is 0 (kafka.server.ZookeeperLeaderElector$LeaderChangeListener) |

\*\* Hostname 이슈 발생시

Error: Exception thrown by the agent : java.net.MalformedURLException: Local host name unknown: java.net.UnknownHostException: monasca-server: monasca-server: Name or service not known

=> /etc/hosts 파일에 아래와 같이 정보 등록

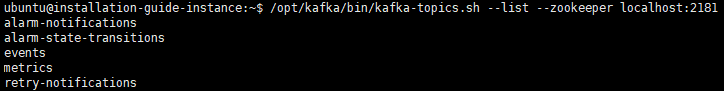
127.0.0.1 localhost “hostname 정보” local Local

1. kafka topic 생성

|  |
| --- |
| **$ /opt/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 32 --topic metrics**  **$ /opt/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 6 --topic events**  **$ /opt/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 6 --topic alarm-state-transitions**  **$ /opt/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 6 --topic alarm-notifications**  **$ /opt/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 3 --topic retry-notifications** |

1. 생성된 topic 리스트 확인

|  |
| --- |
| **$ /opt/kafka/bin/kafka-topics.sh --list --zookeeper localhost:2181** |



1. Apache Storm 설치
2. Apache storm 다운로드

|  |
| --- |
| **$ wget http://apache.mirrors.tds.net/storm/apache-storm-1.0.0/apache-storm-1.0.0.tar.gz** |

1. 압축해제 및 서비스 디렉토리 변경 (Optional)

|  |
| --- |
| **$ tar zxf apache-storm-1.0.0.tar.gz**  **$ mv apache-storm-1.0.0 storm**  **$ sudo mv storm /opt/** |

1. Apache storm supervisor 서비스 시작 스크립트 생성

|  |
| --- |
| **$ sudo vi /etc/init/storm-supervisor.conf**  ---  # Startup script for Storm Supervisor  description "Storm Supervisor daemon"  start on runlevel [2345]  console log  respawn  kill timeout 240  respawn limit 25 5  setgid storm  setuid storm  chdir /opt/storm/  exec /opt/storm/bin/storm supervisor |

1. Apache storm nimbus 서비스 시작 스크립트 생성

|  |
| --- |
| **$ sudo vi /etc/init/storm-nimbus.conf**  ---  # Startup script for Storm Nimbus  description "Storm Nimbus daemon"  start on runlevel [2345]  console log  respawn  kill timeout 240  respawn limit 25 5  setgid storm  setuid storm  chdir /opt/storm/  exec /opt/storm/bin/storm nimbus |

1. apache storm 설정 파일 수정

|  |
| --- |
| **$ sudo vi /opt/storm/conf/storm.yaml**  # 아래 사항을 추가한다.  ---  ### base  java.library.path: "/usr/local/lib:/opt/local/lib:/usr/lib"  storm.local.dir: "/var/storm"  ### zookeeper.\*  storm.zookeeper.servers:  - "localhost"  storm.zookeeper.port: 2181  storm.zookeeper.retry.interval: 5000  storm.zookeeper.retry.times: 29  storm.zookeeper.root: "/storm"  storm.zookeeper.session.timeout: 30000  ### supervisor.\* configs are for node supervisors  supervisor.slots.ports:  - 6701  - 6702  - 6703  - 6704  supervisor.childopts: "-Xmx1024m"  ### worker.\* configs are for task workers  worker.childopts: "-Xmx1280m -XX:+UseConcMarkSweepGC -Dcom.sun.management.jmxremote"  ### nimbus.\* configs are for the masteri  nimbus.host: "localhost"  nimbus.thrift.port: 6627  mbus.childopts: "-Xmx1024m"  ### ui.\* configs are for the master  ui.host: 127.0.0.1  ui.port: 8078  ui.childopts: "-Xmx768m"  ### drpc.\* configs  ### transactional.\* configs  transactional.zookeeper.servers:  - "localhost"  transactional.zookeeper.port: 2181  transactional.zookeeper.root: "/storm-transactional"  ### topology.\* configs are for specific executing storms  topology.acker.executors: 1  topology.debug: false  logviewer.port: 8077  logviewer.childopts: "-Xmx128m" |

1. apache storm 사용자 및 필요한 디렉토리 설정

|  |
| --- |
| **$ sudo useradd storm -U -r**  **$ sudo mkdir /var/storm**  **$ sudo mkdir /opt/storm/logs**  **$ sudo chown -R storm. /var/storm**  **$ sudo chown -R storm. /opt/storm/logs** |

1. apache storm 서비스 시작

|  |
| --- |
| **$ sudo service storm-nimbus start**  **$ sudo service storm-supervisor start** |

1. 확인

|  |
| --- |
| **$ sudo tail -5f /var/log/upstart/storm-nimbus.log**  --- 아래와 같이 정상적인 로그가 보인다면 성공 ---  Running: java -server -Ddaemon.name=nimbus -Dstorm.options= -Dstorm.home=/opt/storm -Dstorm.log.dir=/opt/storm/logs -Djava.library.path=/usr/local/lib:/opt/local/lib:/usr/lib -Dstorm.conf.file= -cp /opt/storm/lib/minlog-1.3.0.jar:/opt/storm/lib/servlet-api-2.5.jar:/opt/storm/lib/storm-rename-hack-1.0.0.jar:/opt/storm/lib/log4j-core-2.1.jar:/opt/storm/lib/asm-5.0.3.jar:/opt/storm/lib/storm-core-1.0.0.jar:/opt/storm/lib/log4j-api-2.1.jar:/opt/storm/lib/kryo-3.0.3.jar:/opt/storm/lib/slf4j-api-1.7.7.jar:/opt/storm/lib/clojure-1.7.0.jar:/opt/storm/lib/log4j-over-slf4j-1.6.6.jar:/opt/storm/lib/log4j-slf4j-impl-2.1.jar:/opt/storm/lib/disruptor-3.3.2.jar:/opt/storm/lib/reflectasm-1.10.1.jar:/opt/storm/lib/objenesis-2.1.jar:/opt/storm/conf -Xmx1024m -Dlogfile.name=nimbus.log -DLog4jContextSelector=org.apache.logging.log4j.core.async.AsyncLoggerContextSelector -Dlog4j.configurationFile=/opt/storm/log4j2/cluster.xml org.apache.storm.daemon.nimbus  **$ sudo tail -5f /var/log/upstart/storm-supervisor.log**  --- 아래와 같이 정상적인 로그가 보인다면 성공 ---  Running: java -server -Ddaemon.name=supervisor -Dstorm.options= -Dstorm.home=/opt/storm -Dstorm.log.dir=/opt/storm/logs -Djava.library.path=/usr/local/lib:/opt/local/lib:/usr/lib -Dstorm.conf.file= -cp /opt/storm/lib/minlog-1.3.0.jar:/opt/storm/lib/servlet-api-2.5.jar:/opt/storm/lib/storm-rename-hack-1.0.0.jar:/opt/storm/lib/log4j-core-2.1.jar:/opt/storm/lib/asm-5.0.3.jar:/opt/storm/lib/storm-core-1.0.0.jar:/opt/storm/lib/log4j-api-2.1.jar:/opt/storm/lib/kryo-3.0.3.jar:/opt/storm/lib/slf4j-api-1.7.7.jar:/opt/storm/lib/clojure-1.7.0.jar:/opt/storm/lib/log4j-over-slf4j-1.6.6.jar:/opt/storm/lib/log4j-slf4j-impl-2.1.jar:/opt/storm/lib/disruptor-3.3.2.jar:/opt/storm/lib/reflectasm-1.10.1.jar:/opt/storm/lib/objenesis-2.1.jar:/opt/storm/conf -Xmx1024m -Dlogfile.name=supervisor.log -DLog4jContextSelector=org.apache.logging.log4j.core.async.AsyncLoggerContextSelector -Dlog4j.configurationFile=/opt/storm/log4j2/cluster.xml org.apache.storm.daemon.supervisor |

1. InfluxDB 설치
2. influxDB repository 등록

|  |
| --- |
| **$ sudo apt-get update**  **$ curl -sL https://repos.influxdata.com/influxdb.key | sudo apt-key add -**  **$ echo "deb https://repos.influxdata.com/ubuntu trusty stable" | sudo tee /etc/apt/sources.list.d/influxdb.list** |

1. influxDB 및 관련 dependencies 설치

|  |
| --- |
| **$ sudo apt-get update**  **$ sudo apt-get install -y influxdb**  **$ sudo apt-get install -y apt-transport-https** |

1. influxDB 서비스 시작

|  |
| --- |
| **$ sudo service influxdb start** |

1. 메트릭스 관련 데이터베이스 생성 및 정책 등록

|  |
| --- |
| **$ influx**  Connected to http://localhost:8086 version 1.3.1  InfluxDB shell version: 1.3.1  > CREATE DATABASE mon  > CREATE USER monasca WITH PASSWORD 'password'  > CREATE RETENTION POLICY persister\_all ON mon DURATION 90d REPLICATION 1 DEFAULT  > quit |

**# Alarm 관련 정보를 관리하기 위한 데이터베이스 생성 및 관리자 정보 등록**

1. 확인

|  |
| --- |
| **$ influx -username monasca -password “password”**  Connected to http://localhost:8086 version 1.3.1  InfluxDB shell version: 1.3.1  > show databases  name: databases  name  ----  mon  \_internal |

1. Monasca Persister 설치
2. monasca persister 설치

|  |
| --- |
| **$ sudo pip install --upgrade pbr**  **$ sudo pip install influxdb**  **$ sudo pip install git+https://git.openstack.org/openstack/monasca-persister@1.6.0#egg=monasca-persister** |

1. persister 사용자 정보 및 디렉토리 등록

|  |
| --- |
| **$ sudo groupadd --system monasca**  **$ sudo useradd --system --gid monasca monasca**  **$ sudo mkdir -p /var/lib/monasca-persister**  **$ sudo mkdir -p /var/log/monasca/persister**  **$ sudo chown monasca:monasca /var/lib/monasca-persister**  **$ sudo chown monasca:monasca /var/log/monasca/persister**  **$ sudo chown root:monasca /etc/monasca/persister.conf**  **$ sudo chmod 640 /etc/monasca/persister.conf** |

1. configuration 파일 생성

|  |
| --- |
| **$ sudo vi /etc/monasca/persister.conf**  ---  [DEFAULT]  log\_config\_append=/etc/monasca/persister-logging.conf  [repositories]  # The driver to use for the metrics repository  metrics\_driver = monasca\_persister.repositories.influxdb.metrics\_repository:MetricInfluxdbRepository  #metrics\_driver = monasca\_persister.repositories.cassandra.metrics\_repository:MetricCassandraRepository  # The driver to use for the alarm state history repository  alarm\_state\_history\_driver = monasca\_persister.repositories.influxdb.alarm\_state\_history\_repository:AlarmStateHistInfluxdbRepository  #alarm\_state\_history\_driver = monasca\_persister.repositories.cassandra.alarm\_state\_history\_repository:AlarmStateHistCassandraRepository  [zookeeper]  # Comma separated list of host:port  uri = **localhost:2181**  partition\_interval\_recheck\_seconds = 15  [kafka\_alarm\_history]  # Comma separated list of Kafka broker host:port.  uri = **localhost:9092**  group\_id = 1\_alarm-state-transitions  topic = alarm-state-transitions  consumer\_id = **consumers**  client\_id = 1  database\_batch\_size = 1000  max\_wait\_time\_seconds = 30  # The following 3 values are set to the kakfa-python defaults  fetch\_size\_bytes = 4096  buffer\_size = 4096  # 8 times buffer size  max\_buffer\_size = 32768  # Path in zookeeper for kafka consumer group partitioning algo  zookeeper\_path = /persister\_partitions/alarm-state-transitions  num\_processors = 1  [kafka\_metrics]  # Comma separated list of Kafka broker host:port  uri = localhost:9092  group\_id = 1\_metrics  topic = metrics  consumer\_id = **consumers**  client\_id = 1  database\_batch\_size = 1000  max\_wait\_time\_seconds = 30  # The following 3 values are set to the kakfa-python defaults  fetch\_size\_bytes = 4096  buffer\_size = 4096  # 8 times buffer size  max\_buffer\_size = 32768  # Path in zookeeper for kafka consumer group partitioning algo  zookeeper\_path = /persister\_partitions/metrics  num\_processors = 1  [influxdb]  database\_name = mon # influxdb 데이터베이스 정보  ip\_address = **localhost** # influxdb 접속 아이피  port = 8086 # influxdb 접속 포트  user = **monasca** # influxdb 사용자 아이디  password = **password** # influxdb 사용자 패스워드 |

1. monasca persister 시작 스크립트 작성

|  |
| --- |
| **$ sudo vi /etc/init/monasca-persister.conf**  ---  # Startup script for the Monasca Persister  description "Monasca Persister Java app"  start on runlevel [2345]  console log  respawn  script  monasca-persister \  --config-file /etc/monasca/persister.conf  end script |

1. monasca persister 시작

|  |
| --- |
| **$ sudo service monasca-persister start** |

1. 확인

|  |
| --- |
| **$ ps -ef |grep monasca-persister** |



1. Monasca Comon 설치
2. monasca common 다운로드

|  |
| --- |
| **$ git clone -b 2.0.0** [**https://github.com/openstack/monasca-common**](https://github.com/openstack/monasca-common)  **$ cd monasca-common** |

1. monasca common 오픈소스 compile and package

|  |
| --- |
| **$ cd java**  **$ mvn clean install** |

1. 확인

|  |
| --- |
| # maven repository에 monasca-common-1.2.1-SNAPSHOPT 이 생성된 것을 확인한다. |

1. Monasca Thresh 설치
2. monasca thresh 다운로드

|  |
| --- |
| **$ git clone -b 1.4.0** [**https://github.com/openstack/monasca-thresh**](https://github.com/openstack/monasca-thresh)  **$ cd monasca-thresh** |

1. monasca thresh 오픈소스 compile and package

|  |
| --- |
| **$ ./run\_maven.sh 1.2.1-SNAPSHOT clean package** |

1. 생성된 monasca thresh package 압축해제 및 configuration 파일 수정

|  |
| --- |
| **$ cd target**  # 생성된 monasca-thres package 파일명에 생성일자가 있어 압축해제 명령어가 실행되지 않는다.  # 생성된 package 명을 monasca-thresh-2.1.1-SNAPSHOT.tar.gz 로 변경한다.  **$ mv monasca-thresh-2.1.1-SNAPSHOT-2017-xx-xxT00:20:08-5c1fd5-tar.tar.gz monasca-thresh-2.1.1-SNAPSHOT.tar.gz**  **$ tar xvzf monasca-thresh-2.1.1-SNAPSHOT.tar.gz**  # 압축해제된 디렉토리도 위와 같이 변경한다.  **$ mv monasca-thresh-2.1.1-SNAPSHOT-2017-xx-xxT00:20:08-5c1fd5 monasca-thresh-2.1.1-SNAPSHOT**  **$ cd monasca-thresh-2.1.1-SNAPSHOT**  **$ cd examples**  **$ mv thresh-config.yml-sample thresh-config.yml**  **$ vi thresh-config.yml**  ---  metricSpoutThreads: 2  metricSpoutTasks: 2  statsdConfig:  host: localhost  port: 8125  prefix: monasca.storm.  dimensions: !!map  service : monitoring  component : storm  metricSpoutConfig:  kafkaConsumerConfiguration:  # See http://kafka.apache.org/documentation.html#api for semantics and defaults.  topic: metrics  numThreads: 1  groupId: thresh-metric  zookeeperConnect: localhost:2181  consumerId: 1  socketTimeoutMs: 30000  socketReceiveBufferBytes : 65536  fetchMessageMaxBytes: 1048576  autoCommitEnable: true  autoCommitIntervalMs: 60000  queuedMaxMessageChunks: 10  rebalanceMaxRetries: 4  fetchMinBytes: 1  fetchWaitMaxMs: 100  rebalanceBackoffMs: 2000  refreshLeaderBackoffMs: 200  autoOffsetReset: largest  consumerTimeoutMs: -1  clientId : 1  zookeeperSessionTimeoutMs : 60000  zookeeperConnectionTimeoutMs : 60000  zookeeperSyncTimeMs: 2000  eventSpoutConfig:  kafkaConsumerConfiguration:  # See http://kafka.apache.org/documentation.html#api for semantics and defaults.  topic: events  numThreads: 1  groupId: thresh-event  zookeeperConnect: localhost:2181  consumerId: 1  socketTimeoutMs: 30000  socketReceiveBufferBytes : 65536  fetchMessageMaxBytes: 1048576  autoCommitEnable: true  autoCommitIntervalMs: 60000  queuedMaxMessageChunks: 10  rebalanceMaxRetries: 4  fetchMinBytes: 1  fetchWaitMaxMs: 100  rebalanceBackoffMs: 2000  refreshLeaderBackoffMs: 200  autoOffsetReset: largest  consumerTimeoutMs: -1  clientId : 1  zookeeperSessionTimeoutMs : 60000  zookeeperConnectionTimeoutMs : 60000  zookeeperSyncTimeMs: 2000  kafkaProducerConfig:  # See http://kafka.apache.org/documentation.html#api for semantics and defaults.  topic: alarm-state-transitions  metadataBrokerList: localhost:9092  serializerClass: kafka.serializer.StringEncoder  partitionerClass:  requestRequiredAcks: 1  requestTimeoutMs: 10000  producerType: sync  keySerializerClass:  compressionCodec: none  compressedTopics:  messageSendMaxRetries: 3  retryBackoffMs: 100  topicMetadataRefreshIntervalMs: 600000  queueBufferingMaxMs: 5000  queueBufferingMaxMessages: 10000  queueEnqueueTimeoutMs: -1  batchNumMessages: 200  sendBufferBytes: 102400  clientId : Threshold\_Engine  sporadicMetricNamespaces:  - foo  database:  driverClass: com.mysql.jdbc.Driver  url: jdbc:mysql://localhost/mon?useSSL=true # mysql 접속 정보  user: **monasca** # mysql 사용자 아이디  password: **password** # mysql 사용자 패스워드  properties:  ssl: false  # the maximum amount of time to wait on an empty pool before throwing an exception  maxWaitForConnection: 1s  # the SQL query to run when validating a connection's liveness  validationQuery: "/\* MyService Health Check \*/ SELECT 1"  # the minimum number of connections to keep open  minSize: 8  # the maximum number of connections to keep open  maxSize: 41 |

1. monasca thresh configuration 및 package 파일 이동

|  |
| --- |
| **$ sudo mv thresh-config.yml /etc/monasca/**  **$ cd ..**  **$ mv monasca-thresh.jar /etc/monasca/** |

1. monasca thresh 서비스 시작 스크립트 생성

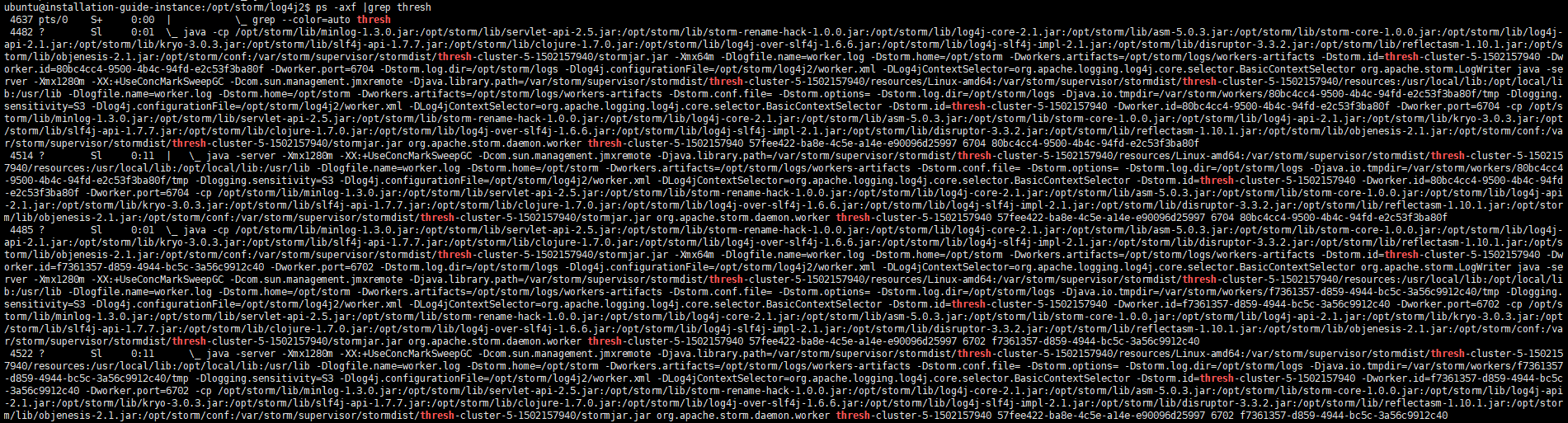
|  |
| --- |
| **$ sudo vi /etc/init.d/monasca-thresh**  ---  #!/bin/bash  #  # (C) Copyright 2015 Hewlett Packard Enterprise Development Company LP  #  # Licensed under the Apache License, Version 2.0 (the "License");  # you may not use this file except in compliance with the License.  # You may obtain a copy of the License at  #  #    http://www.apache.org/licenses/LICENSE-2.0  #  # Unless required by applicable law or agreed to in writing, software  # distributed under the License is distributed on an "AS IS" BASIS,  # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or  # implied.  # See the License for the specific language governing permissions and  # limitations under the License.  #  ### BEGIN INIT INFO  # Provides: monasca-thresh  # Required-Start: $nimbus  # Required-Stop:  # Default-Start: 2 3 4 5  # Default-Stop:  # Short-Description: Monitoring threshold engine running under storm  # Description:  ### END INIT INFO  case "$1" in  start)  $0 status  if [ $? -ne 0 ]; then  sudo -Hu monasca /opt/storm/bin/storm jar /etc/monasca/monasca-thresh.jar monasca.thresh.ThresholdingEngine /etc/monasca/thresh-config.yml thresh-cluster  exit $?  else  echo "monasca-thresh is already running"  exit 0  fi  ;;  stop)  # On system shutdown storm is being shutdown also and this will hang so skip shutting down thresh in that case  if [ -e '/sbin/runlevel' ]; then # upstart/sysV case  if [ $(runlevel | cut -d\ -f 2) == 0 ]; then  exit 0  fi  else # systemd case  systemctl list-units --type=target |grep shutdown.target  if [ $? -eq 0 ]; then  exit 0  fi  fi  sudo -Hu monasca /opt/storm/bin/storm kill thresh-cluster  # The above command returns but actually takes awhile loop watching status  while true; do  sudo -Hu monasca /opt/storm/bin/storm list |grep thresh-cluster  if [ $? -ne 0 ]; then break; fi  sleep 1  done  ;;  status)  sudo -Hu monasca /opt/storm/bin/storm list |grep thresh-cluster  ;;  restart)  $0 stop  $0 start  ;;  esac |

1. monasca thresh 서비스 시작

|  |
| --- |
| **$ sudo chmod +x /etc/init.d/monasca-thresh**  **$ sudo service monasca-thresh start** |

1. 확인

|  |
| --- |
| **$ ps -ef |grep thresh** |



1. Monasca Notification 설치
2. monasca notification 및 dependencies 설치

|  |
| --- |
| **$ sudo pip install git+https://github.com/openstack/monasca-notification@1.9.0**  **$ sudo apt-get install sendmail** |

1. monasca notificatioin 설정 파일 생성

|  |
| --- |
| **$ sudo vi /etc/monasca/notification.yaml**  ---  kafka:  url: 127.0.0.1:9092 # or comma seperated list of multiple hosts  group: monasca-notification  alarm\_topic: alarm-state-transitions  notification\_topic: alarm-notifications  notification\_retry\_topic: retry-notifications  periodic:  60: 60-seconds-notifications  max\_offset\_lag: 600 # In seconds, undefined for none  database:  repo\_driver: monasca\_notification.common.repositories.mysql.mysql\_repo:MysqlRepo  mysql:  host: 127.0.0.1 # mysql 접속 IP  port: 3306 # mysql 접속 port  user: monasca # mysql 사용자 아이디  passwd: password # mysql 사용자 패스워드  db: mon # mysql database 이름  # A dictionary set according to the params defined in, http://dev.mysql.com/doc/refman/5.0/en/mysql-ssl-set.html  # ssl: {'ca': '/path/to/ca'}  notification\_types:  email:  server: 127.0.0.1  port: 25  user:  password:  timeout: 60  from\_addr: ihocho@crossent.com  webhook:  timeout: 5  processors:  alarm:  number: 2  ttl: 14400 # In seconds, undefined for none. Alarms older than this are not processed  notification:  number: 4  retry:  interval: 30  max\_attempts: 5  queues:  alarms\_size: 256  finished\_size: 256  notifications\_size: 256  sent\_notifications\_size: 50 # limiting this size reduces potential # of re-sent notifications after a failure  zookeeper:  url: 127.0.0.1:2181 # or comma seperated list of multiple hosts  notification\_path: /notification/alarms  notification\_retry\_path: /notification/retry  periodic\_path:  60: /notification/60\_seconds  logging: # Used in logging.dictConfig  version: 1  disable\_existing\_loggers: False  formatters:  default:  format: "%(asctime)s %(levelname)s %(name)s %(message)s"  handlers:  console:  class: logging.StreamHandler  formatter: default  file:  class : logging.handlers.RotatingFileHandler  filename: /var/log/monasca/notification/notification.log  formatter: default  maxBytes: 10485760 # Rotate at file size ~10MB  backupCount: 5 # Keep 5 older logs around  loggers:  kazoo:  level: WARN  kafka:  level: WARN  statsd:  level: WARN  root:  handlers:  [console, file]  # - file  level: WARN  statsd:  host: 'localhost'  port: 8125 |

1. monasca notification 시작 스크립트 생성

|  |
| --- |
| **$ sudo vi /etc/init/monasca-notification.conf**  ---  # Startup script for the monasca\_notification  description "Monasca Notification daemon"  start on runlevel [2345]  console log  respawn  setgid monasca  setuid monasca  exec /usr/bin/python /usr/local/bin/monasca-notification |

1. monasca notification 로그 디렉토리 생성

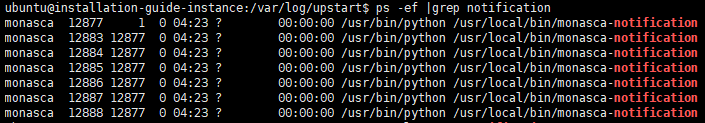
|  |
| --- |
| **$ sudo mkdir -p /var/log/monasca/notification**  **$ sudo chown -R monasca. /var/log/monasca/notification** |

1. monasca notification 서비스 가동

|  |
| --- |
| **$ sudo service monasca-notification start** |

1. 확인

|  |
| --- |
| **$ ps -ef |grep notification** |



1. Monasca API 설치
2. monasca api 다운로드

|  |
| --- |
| **$ git clone -b 2.0.0** [**https://github.com/openstack/monasca-api**](https://github.com/openstack/monasca-api)  **$ cd monasca-api** |

1. run\_maven.sh 파일 수정

|  |
| --- |
| ---  #!/bin/bash  set -x  env  # Download maven 3 if the system maven isn't maven 3  VERSION=`mvn -v | grep "Apache Maven 3"`  if [ -z "${VERSION}" ]; then  curl http://archive.apache.org/dist/maven/binaries/apache-maven-3.2.1-bin.tar.gz > apache-maven-3.2.1-bin.tar.gz  tar -xvzf apache-maven-3.2.1-bin.tar.gz  MVN=${PWD}/apache-maven-3.2.1/bin/mvn  else  MVN=mvn  fi  # Get the expected common version  COMMON\_VERSION=$1  # Get rid of the version argument  shift  # Get rid of the java property name containing the args  shift  RUN\_BUILD=false  for ARG in $\*; do  if [ "$ARG" = "package" ]; then  RUN\_BUILD=true  fi  if [ "$ARG" = "install" ]; then  RUN\_BUILD=true  fi  done  if [ $RUN\_BUILD = "true" ]; then  if [ ! -z "$ZUUL\_BRANCH" ]; then  BRANCH=${ZUUL\_BRANCH}  else  BRANCH=${ZUUL\_REF}  fi  ( cd common; ./build\_common.sh ${MVN} ${COMMON\_VERSION} **2.0.0** )  RC=$?  if [ $RC != 0 ]; then  exit $RC  fi  fi  # Invoke the maven 3 on the real pom.xml  ( cd java; ${MVN} **-Dmaven.test.skip=true** -DgitRevision=`git rev-list HEAD --max-count 1 --abbrev=0 --abbrev-commit` $\* )  RC=$?  # Copy the jars where the publisher will find them  if [ $RUN\_BUILD = "true" ]; then  if [ ! -L target ]; then  ln -sf java/target target  fi  fi  rm -fr apache-maven-3.2.1\*  exit $RC |

1. common/build\_common.sh 파일 수정

|  |
| --- |
| ---  #!/bin/sh  set -x  ME=`whoami`  echo "Running as user: $ME"  MVN=$1  VERSION=$2  BRANCH=$3  check\_user() {  ME=$1  if [ "${ME}" != "jenkins" ]; then  echo "\nERROR: Download monasca-common and do a mvn install to install the monasca-commom jars\n" 1>&2  exit 1  fi  }  BUILD\_COMMON=false  POM\_FILE=~/.m2/repository/monasca-common/monasca-common/${VERSION}/monasca-common-${VERSION}.pom  if [ ! -r "${POM\_FILE}" ]; then  check\_user ${ME}  BUILD\_COMMON=true  fi  # This should only be done on the stack forge system  if [ "${BUILD\_COMMON}" = "true" ]; then  git clone -b ${BRANCH} https://git.openstack.org/openstack/monasca-common  cd monasca-common  ${MVN} clean  ${MVN} install **-Dmaven.test.skip=true**  fi |

1. monasca api 소스 compile & package

|  |
| --- |
| **$ ./run\_maven.sh 1.2.1-SNAPSHOT clean package** |

1. monasca api package 파일 압축 해제 및 configuration 파일 수정

|  |
| --- |
| **$ cd target**  **$ tar xvzf monasca-api-1.2.1-SNAPSHOT-tar.tar.gz**  **$ cd monasca-api-1.2.1-SNAPSHOT/**  **$ cd examples**  **$ mv api-config.yml-sample api-config.yml**  **$ vi api-config.yml**  ---  # The region for which all metrics passing through this server will be persisted  region: **RegionOne** # Region 이름  # Maximum rows (Mysql) or points (Influxdb) to return when listing elements  maxQueryLimit: 10000  # Whether this server is running on a secure port  accessedViaHttps: false  # Topic for publishing metrics to  metricsTopic: metrics  # Topic for publishing domain events to  eventsTopic: events  **validNotificationPeriods:**  **- 60**  kafka:  brokerUris:  - **localhost:9092** # kafka 접속 정보  zookeeperUris:  - **localhost:2181** # zookeeper 접속 정보  healthCheckTopic: healthcheck  mysql:  driverClass: com.mysql.jdbc.Driver  url: **jdbc:mysql://localhost:3306/mon?connectTimeout=5000&autoReconnect=true&useLegacyDatetimeCode=false** # mysql 접속 정보  user: **monasca** # mysql 사용자 아이디  password: **password**  # mysql 사용자 패스워드  maxWaitForConnection: 1s  validationQuery: "/\* MyService Health Check \*/ SELECT 1"  minSize: 8  maxSize: 32  checkConnectionWhileIdle: false  checkConnectionOnBorrow: true  databaseConfiguration:  databaseType: **influxdb**  influxDB:  version: V9  maxHttpConnections: 100  # Retention policy may be left blank to indicate default policy.  retentionPolicy:  name: **mon** # influxdb database 이름  url: <http://localhost:8086># influxdb http 접속 정보  user: **monasca** # influxdb 사용자 아이디  password: **password** # influxdb 사용자 패스워드  vertica:  driverClass: com.vertica.jdbc.Driver  url: jdbc:vertica://192.168.10.8/mon  user: dbadmin  password: password  maxWaitForConnection: 1s  validationQuery: "/\* MyService Health Check \*/ SELECT 1"  minSize: 4  maxSize: 32  checkConnectionWhileIdle: false  #  # vertica database hint to be added to SELECT  # statements. For example, the hint below is used  # to tell vertica that the query can be satisfied  # locally (replicated projection).  #  # dbHint: "/\*+KV(01)\*/"  dbHint: ""  middleware:  enabled: true  serverVIP: **xxx.xxx.xxx.xxx** #keystone ip 정보  serverPort: **35357** #keystone 인증 port  useHttps: false  truststore: "**None**"  truststorePassword: "**None**"  connTimeout: 500  connSSLClientAuth: false  keystore: "**None**"  keystorePassword: **false**  connPoolMaxActive: 3  connPoolMaxIdle: 3  connPoolEvictPeriod: 600000  connPoolMinIdleTime: 600000  connRetryTimes: 2  connRetryInterval: 50  defaultAuthorizedRoles: [**admin**, user, domainuser, domainadmin, monasca-user]  readOnlyAuthorizedRoles: [**admin**, monasca-read-only-user]  agentAuthorizedRoles: [**monitoring-delegate**] #cross-tenant role 정보  adminAuthMethod: **password** #사용자 인증 방식  adminUser: **monasca-agent** #cross-tenant 사용자 아이디  adminPassword: **cfmonit**  #cross-tenant 사용자 패스워드  adminProjectId: **9c1a27e20412473b843dbf32bdec2390** #관리 Project guid 정보  adminProjectName: "**admin**" #관리 Project 이름  adminUserDomainId: **9c6e016d8b3642109655740c26e5eb57** #domain guid 정보  adminUserDomainName: **9c6e016d8b3642109655740c26e5eb57** #domain guid 정보  adminProjectDomainId:  adminProjectDomainName:  adminToken:  timeToCacheToken: 600  maxTokenCacheSize: 1048576  server:  applicationConnectors:  - type: http  port: **8020** # monasca api listen port  maxRequestHeaderSize: 16KiB # Allow large headers used by keystone tokens  requestLog:  timeZone: UTC  appenders:  - type: file  currentLogFilename: /var/log/monasca/api/request.log  threshold: ALL  archive: true  archivedLogFilenamePattern: /var/log/monasca/api/request-%d.log.gz  archivedFileCount: 5  # Logging settings.  logging:  # The default level of all loggers. Can be OFF, ERROR, WARN, INFO, DEBUG, TRACE, or ALL.  level: **WARN** # 로그 레벨 설정  # Logger-specific levels.  loggers:  # Sets the level for 'com.example.app' to DEBUG.  com.example.app: DEBUG  appenders:  - type: console  threshold: ALL  timeZone: UTC  target: stdout  logFormat: # TODO  - type: file  currentLogFilename: /var/log/monasca/api/monasca-api.log  threshold: ALL  archive: true  archivedLogFilenamePattern: /var/log/monasca/api/monasca-api-%d.log.gz  archivedFileCount: 5  timeZone: UTC  logFormat: # TODO  - type: syslog  host: localhost  port: 514  facility: local0  threshold: ALL |

1. monasca api package 파일 및 configuration 파일 이동 (optional)

|  |
| --- |
| # 7단계 monasca api 서비스 시작 스크립트에서 참조하는 monasca-api.jar 및 api-config.yml 파일의 위치를 관리하기 손쉬운 곳으로 이동시킨다.  $ mv ~/where-at-monasca-api-directory/target/monasca-api-1.2.1-SNAPSHOT/monasca-api.jar ~/monasca-api/  $ mv ~/where-at-monasca-api-directory/target/monasca-api-1.2.1-SNAPSHOT/examples/api-config.yml ~/monasca-api/ |

1. monasca api 서비스 시작 스크립트 생성

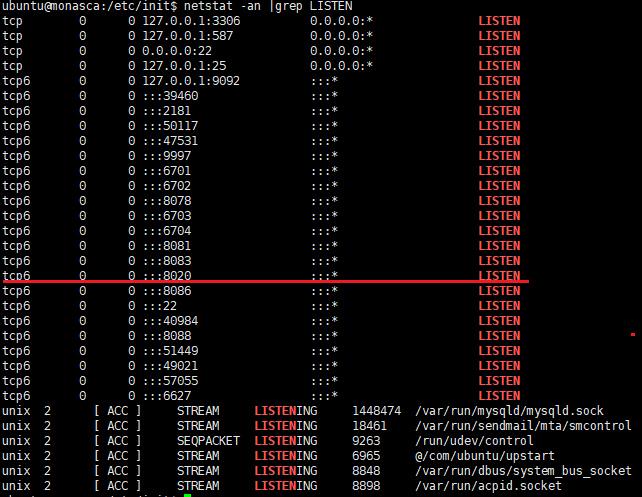
|  |
| --- |
| **$ sudo vi /etc/init/monasca-api.conf**  ---  # Startup script for the monasca\_api  description "Monasca Notification daemon"  start on runlevel [2345]  console log  respawn  setgid monasca  setuid monasca  exec java -jar **/home/ubuntu/monasca-api/monasca-api.jar** server **/home/ubuntu/monasca-api/api-config.yml** |

1. monasca api 서비스 시작

|  |
| --- |
| **$ sudo service monasca-api start** |

1. 확인

|  |
| --- |
| **$ netstat -an |grep LISTEN** |



1. Elasticsearch 관련 프로그램 설치
   1. Elasticserarch 서버 설치
2. dependencies 설치

|  |
| --- |
| **$ sudo apt-get update**  **$ sudo apt-get install -y python-software-properties software-properties-common** |

1. Elasticsearch repository 등록

|  |
| --- |
| **$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -**  **$ echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-5.x.list** |

1. Elasticsearch 설치

|  |
| --- |
| **$ sudo apt-get update**  **$ sudo apt-get install -y elasticsearch** |

1. 사용자 그룹 추가 - Elasticsearch

|  |
| --- |
| **$ sudo usermod -a -G elasticsearch “사용자 계정”** |

1. Elasticsearch configuration 파일 수정

|  |
| --- |
| **$ cd /etc/elasticsearch && sudo vi elasticsearch.yml**  ---  ...  # Lock the memory on startup:  **bootstrap.memory\_lock: true**  **...**  # Set the bind address to a specific IP (IPv4 or IPv6):  **network.host: localhost**  # Set a custom port for HTTP:  **http.port: 9200**  ... |

1. Elasticsearch service 파일 수정

|  |
| --- |
| **$ sudo vi /usr/lib/systemd/system/elasticsearch.service**  ---  ...  # Specifies the maximum number of bytes of memory that may be locked into RAM  # Set to "infinity" if you use the 'bootstrap.memory\_lock: true' option  # in elasticsearch.yml and 'MAX\_LOCKED\_MEMORY=unlimited' in /etc/default/elasticsearch  **LimitMEMLOCK=infinity**  ... |

1. Elasticsearch default 파일 수정

|  |
| --- |
| **$ sudo vi /etc/default/elasticsearch**  ---  ...  # The maximum number of bytes of memory that may be locked into RAM  # Set to "unlimited" if you use the 'bootstrap.memory\_lock: true' option  # in elasticsearch.yml.  # When using Systemd, the LimitMEMLOCK property must be set  # in /usr/lib/systemd/system/elasticsearch.service  **MAX\_LOCKED\_MEMORY=unlimited**  ... |

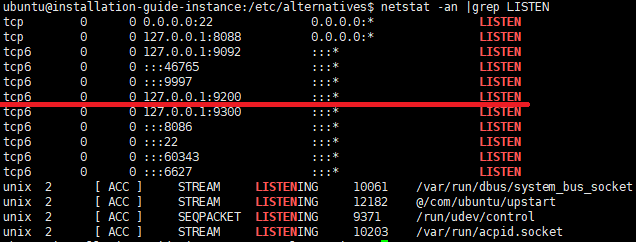
1. Elasticsearch 서비스 시작

|  |
| --- |
| **$ sudo service elasticsearch start** |

1. 확인

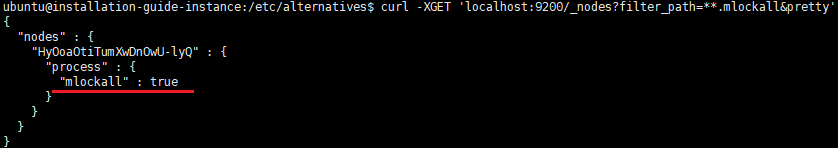
9.1 Elasticserarch 서버 가동 여부

|  |
| --- |
| **$ netstat -plntu** |



9.2 mlockall 정보가 “enabled” 되었는지 확인

|  |
| --- |
| **$ curl -XGET 'localhost:9200/\_nodes?filter\_path=\*\*.mlockall&pretty'** |



* 1. logstash 설치

1. logstash 설치

|  |
| --- |
| **$ sudo apt-get install -y logstash** |

1. /etc/hosts 파일 수정

|  |
| --- |
| **$ sudo vi /etc/hosts**  ---  “private network ip” “hostname”  ex) 10.244.2.22 installation-guide-server |

1. SSL certificate 파일 생성

|  |
| --- |
| **$ cd /etc/logstash**  **$ sudo openssl req -subj /CN=”hostaname” -x509 -days 3650 -batch -nodes -newkey rsa:4096 -keyout logstash.key -out logstash.crt** |

1. filebeat-input.conf 파일 생성

|  |
| --- |
| **$ cd /etc/logstash**  **$ sudo vi conf.d/filebeat-input.conf** ---  input {  beats {  port => 5443 #filebeat 정보를 수신하기 위한 Listen port  type => syslog  ssl => true  ssl\_certificate => "/etc/logstash/logstash.crt"  ssl\_key => "/etc/logstash/logstash.key"  }  } |

1. syslog-filter.conf 파일 생성

|  |
| --- |
| **$ cd /etc/logstash**  **$ sudo vi conf.d/syslog-filter.conf** ---  filter {  if [type] == "syslog" {  grok {  match => { "message" => "%{SYSLOGTIMESTAMP:syslog\_timestamp} %{SYSLOGHOST:syslog\_hostname} %{DATA:syslog\_program}(?:\[%{POSINT:syslog\_pid}\])?: %{GREEDYDATA:syslog\_message}" }  add\_field => [ "received\_at", "%{@timestamp}" ]  add\_field => [ "received\_from", "%{host}" ]  }  date {  match => [ "syslog\_timestamp", "MMM d HH:mm:ss", "MMM dd HH:mm:ss" ]  }  }  } |

1. output-elasticsearch.conf 파일 생성

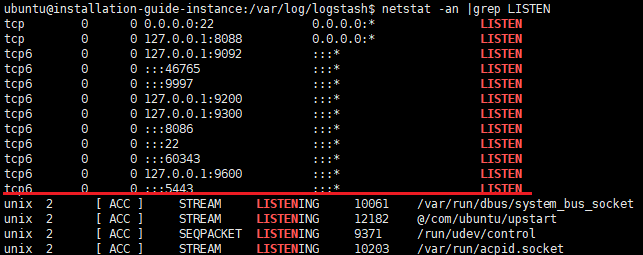
|  |
| --- |
| **$ cd /etc/logstash**  **$ sudo vi conf.d/output-elasticsearch.conf** ---  output {  elasticsearch { hosts => ["**”your elastic ip**”:9200"] # 설치된 환경의 IP 정보  hosts => "”**your elastic ip**”:9200" # 설치된 환경의 IP 정보  manage\_template => false  index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"  document\_type => "%{[@metadata][type]}"  }  } |

1. logstash 서비스 시작 파일 생성

|  |
| --- |
| **$ sudo service logstash start** |

1. 확인

|  |
| --- |
| **$ netstat -an |grep LISTEN** |



* Reference : Cross-Project(Tenant) 사용자 추가 및 권한 부여

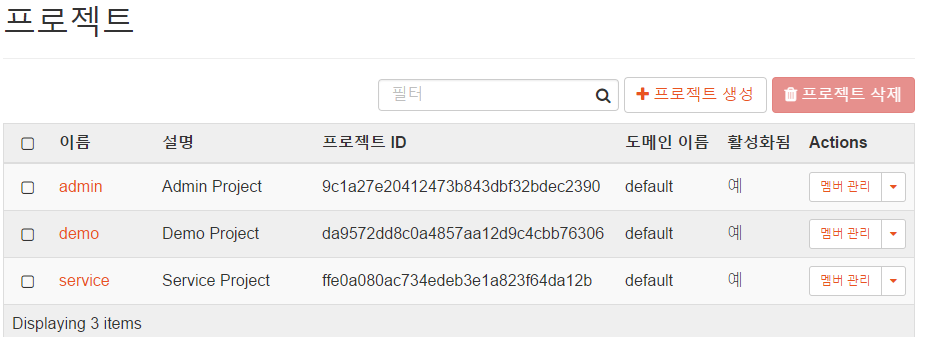
Openstack 기반으로 생성된 모든 Project(Tenant)의 정보를 하나의 계정으로 수집 및 조회하기 위해서는 Cross-Tenant 사용자를 생성하여, 각각의 Project(Tenant)마다 조회할 수 있도록 멤버로 등록한다.

Openstack Cli를 이용하여 Cross-Tenant 사용자를 생성한 후, Openstack Horizon 화면으로 통해 각각의 프로젝트 사용자 정보에 생성한 Cross-Tenant 사용자 및 권한을 부여한다.

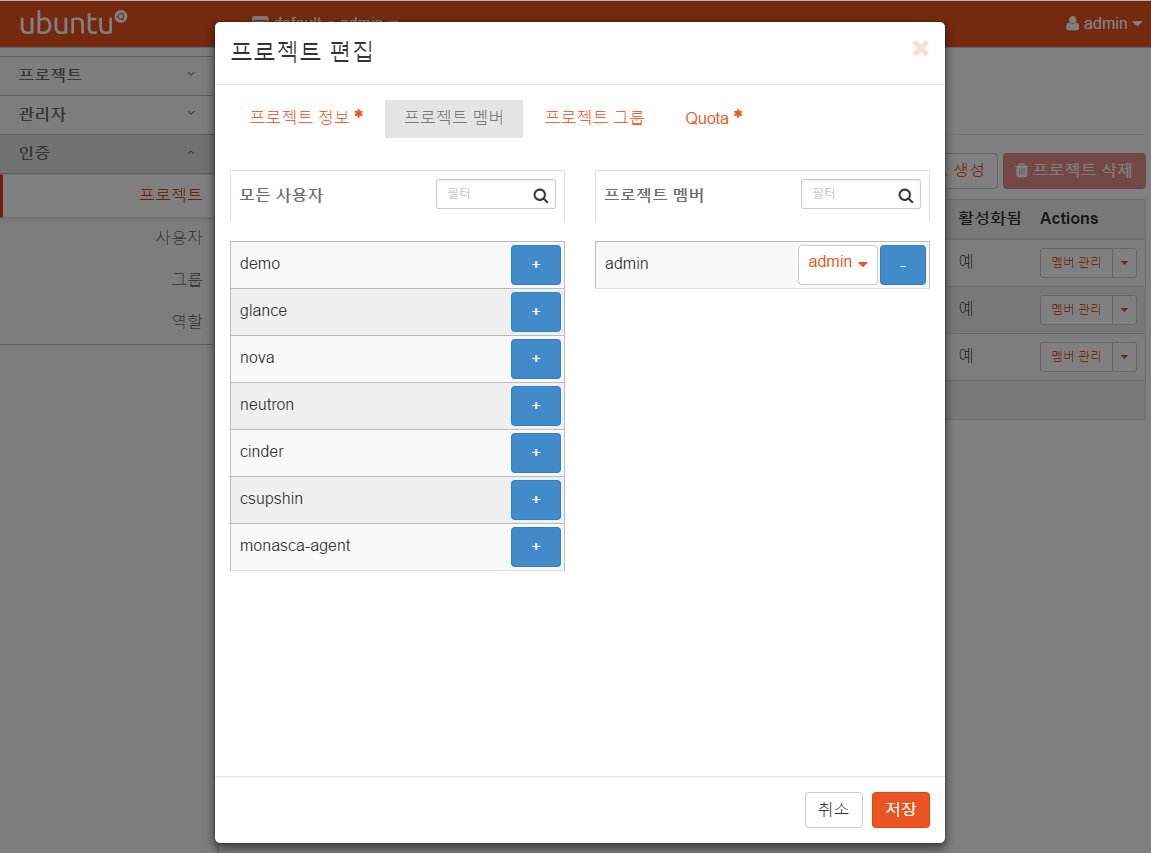
1. Cross-Tenant 사용자 생성

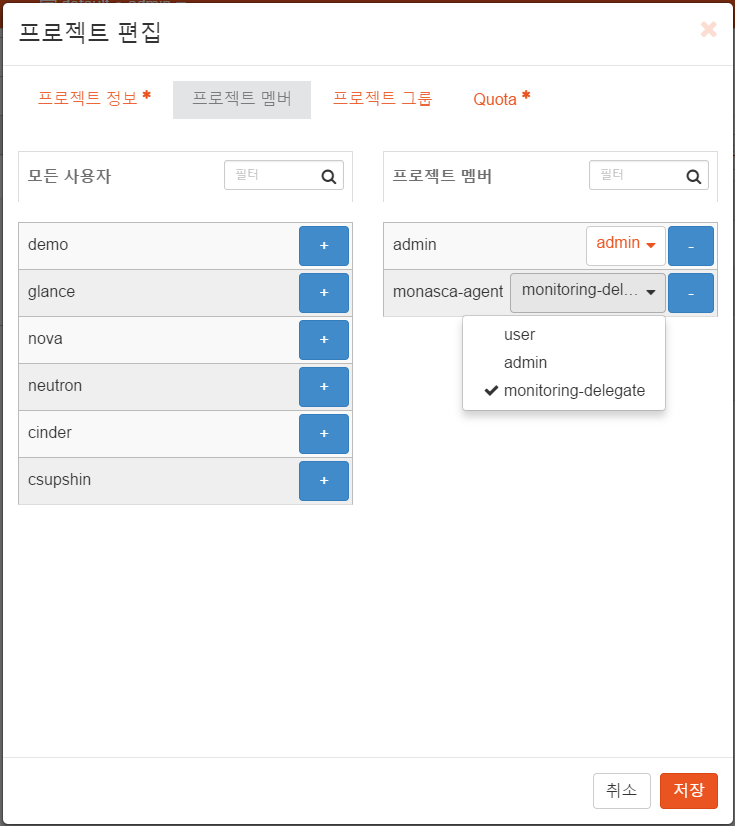
|  |
| --- |
| $ openstack user create --domain default --password-prompt monasca-agent  $ openstack role create monitoring-delegate |

1. Project 사용자 추가



# 각각의 프로젝트 멤버관리에 추가한 Cross-Tenant 사용자 정보를 등록한다.





# 추가한 Cross-Tenant 사용자를 선택 후, 생성한 Role을 지정한다.

1. 변경 내용: 변경이 발생되는 위치와 변경 내용을 자세히 기록(장/절과 변경 내용을 기술한다.) [↑](#footnote-ref-1)