KOSMO KOREA SMART MANUFACTURING OFFICE

NEST+ FIELD

에지 게이트웨이 솔루션

에지 게이트웨이 설치 및 운영

2022. 02. 01

스마트제조혁신추진단 X 네스트필드㈜

KOSMO KOREA SMART MANUFACTURING OFFICE

보고순서

KOREA SMART
MANUFACTURING OFFICE

KOSMO KOREA SMART MANUFACTURING OFFICE

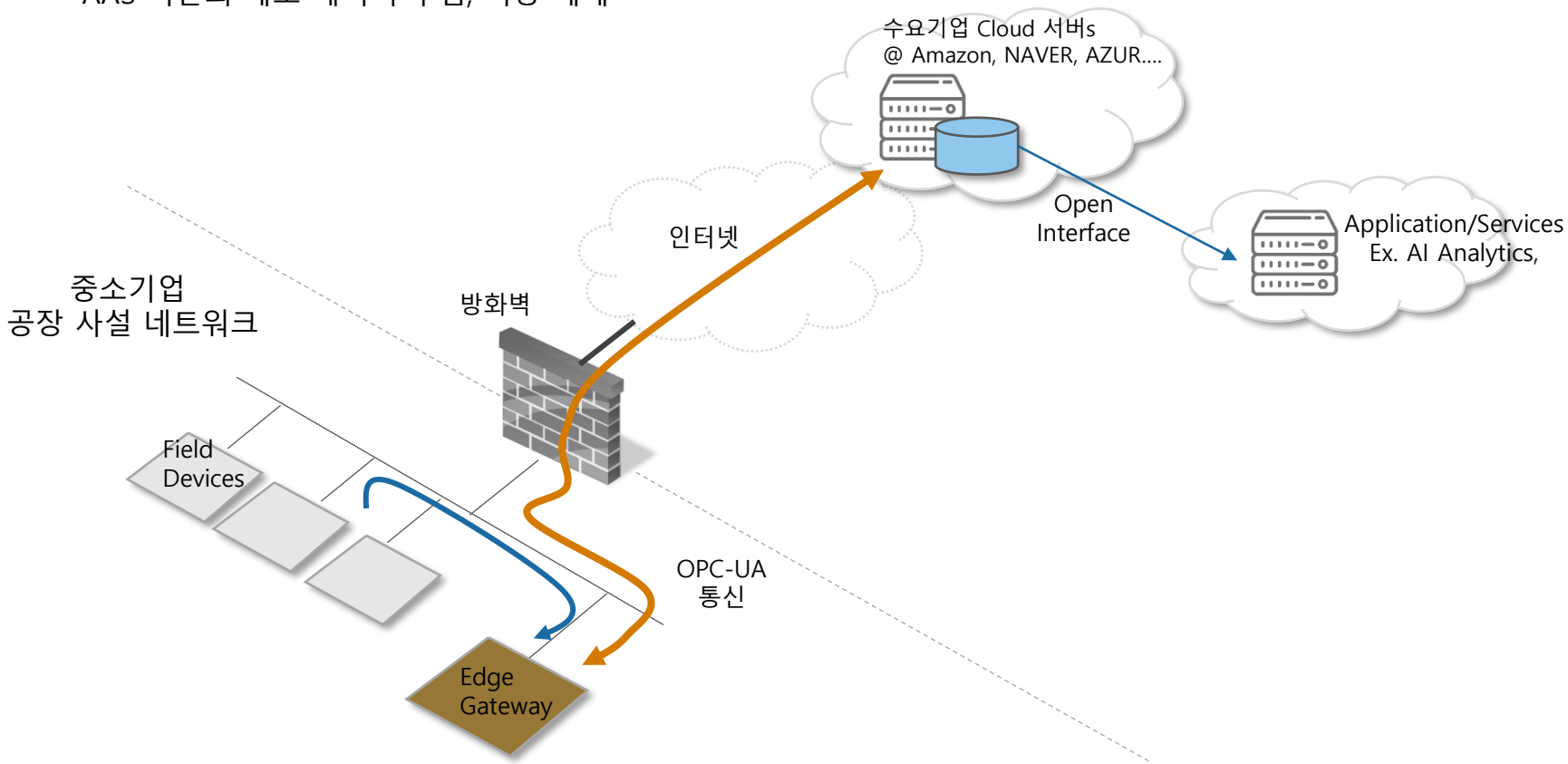
01

에지 게이트웨이 소개

KOREA SMART MANUFACTURING OFFICE

01. 에지 게이트웨이 소개

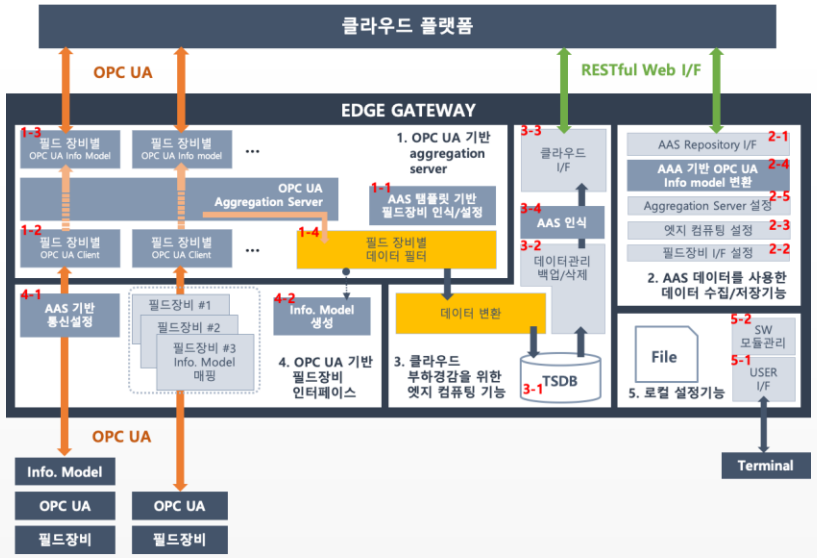
AAS 기반의 제조 데이터 수집, 저장 체계



공장내 PLC등 여러 필드장치로부터 데이터를 수집하여
Edge Gateway 내부에 저장하고
OPC-UA 표준 통신방법으로 데이터를 클라우드 서버로 전달

01. 에지 게이트웨이 소개

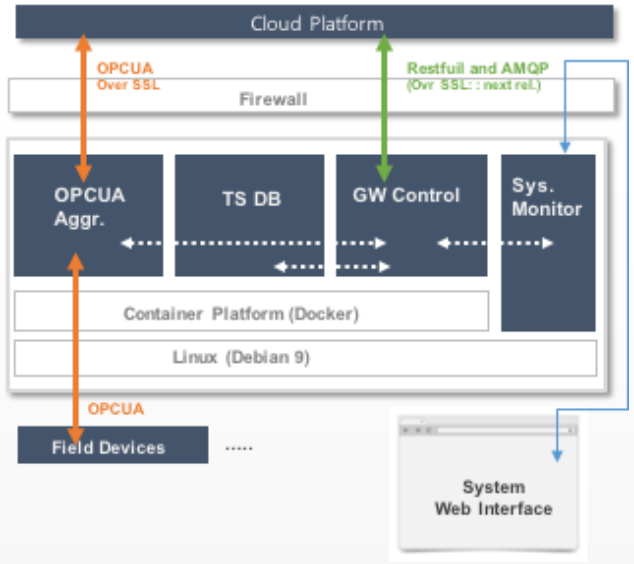
게이트웨이 시스템 기능



에지 게이트웨이 시스템의 기본 기능 정의

- OPC UA 클라이언트 : 필드장비인터페이스
- OPC UA 서버 : Aggregation & 클라우드 인터페이스
- 데이터의 시계열 DB 저장 : 에지컴퓨팅 지원
- 시스템의 설정 및 관리 인터페이스

게이트웨이 시스템 구조



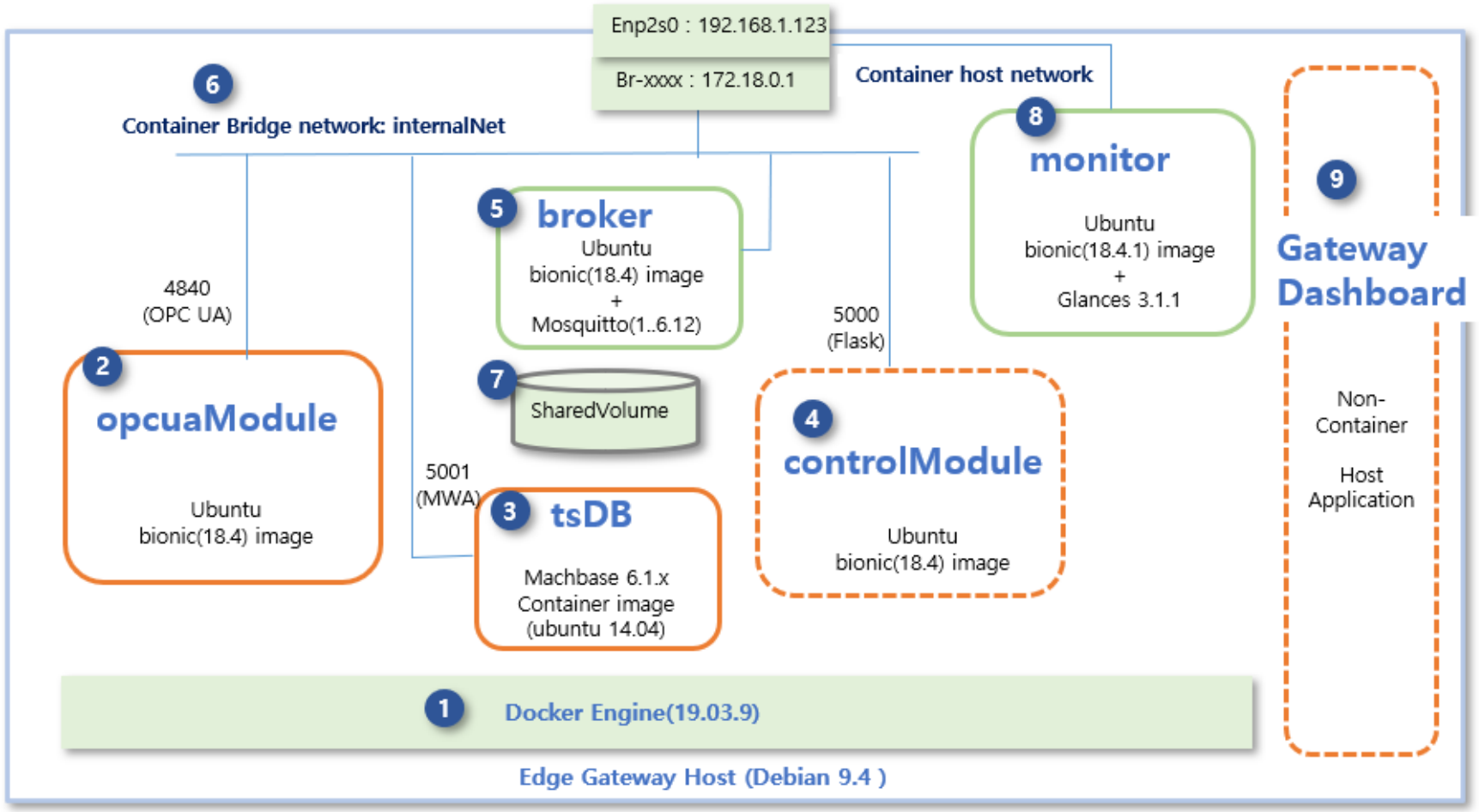
에지 게이트웨이 시스템 개발

- Open 프레임워크를 이용한 개발
  - Linux, Docker, Restful API, MQTT/AMQP ..
- Container방식의 시스템 모듈화/확장성
- 오픈 플랫폼을 통한 모듈 관리 및 배포 : Docker Hub
- 시스템 모니터링 웹인터페이스

01. 에지 게이트웨이 소개

◆ 에지 게이트웨이 시스템 구성-1

게이트웨이 구성 및 어플리케이션



01. 에지 게이트웨이 소개

◆ 에지 게이트웨이 시스템 구성-2

게이트웨이 구성 및 어플리케이션 (계속)

1 Docker Engine

각 어플리케이션 컨테이너들이 동작하는 플랫폼
컨테이너, 네트워크, 공유볼륨을 포함

2 컨테이너 opcuaModule

게이트웨이의 핵심 어플리케이션인 Aggregation Server가 구동되며 field device로부터 데이터를 수집하여 OPC UA통신으로 클라우드상의 서버로 전달함

3 컨테이너 tsDB

Aggregation Server에 의해 수집되는 필드장비의 데이터가 저장되는 로컬 시계열 Database가 동작
(상용 DB 마크베이스를 이용)

4 컨테이너 controlModule

게이트웨이의 클라우드 등록, 데이터 저장 그리고 시스템의 동작을 관할하는 control 어플리케이션이 구동

5 컨테이너 broker

각 컨테이너의 어플리케이션의 데이터, 명령, 이벤트 송수신을 중개하는 MQTT 브로커가 동작
(open 소스 MQTT broker를 이용)

6 도커 네트워크

각 컨테이너가 통신하는 도커내부 네트워크

7 도커 공유볼륨

각 컨테이너가 디렉토리로서 공유하는 공유저장소. Config파일 접근, 컨테이너간 파일 전달을 위해 사용됨

8 컨테이너 monitor

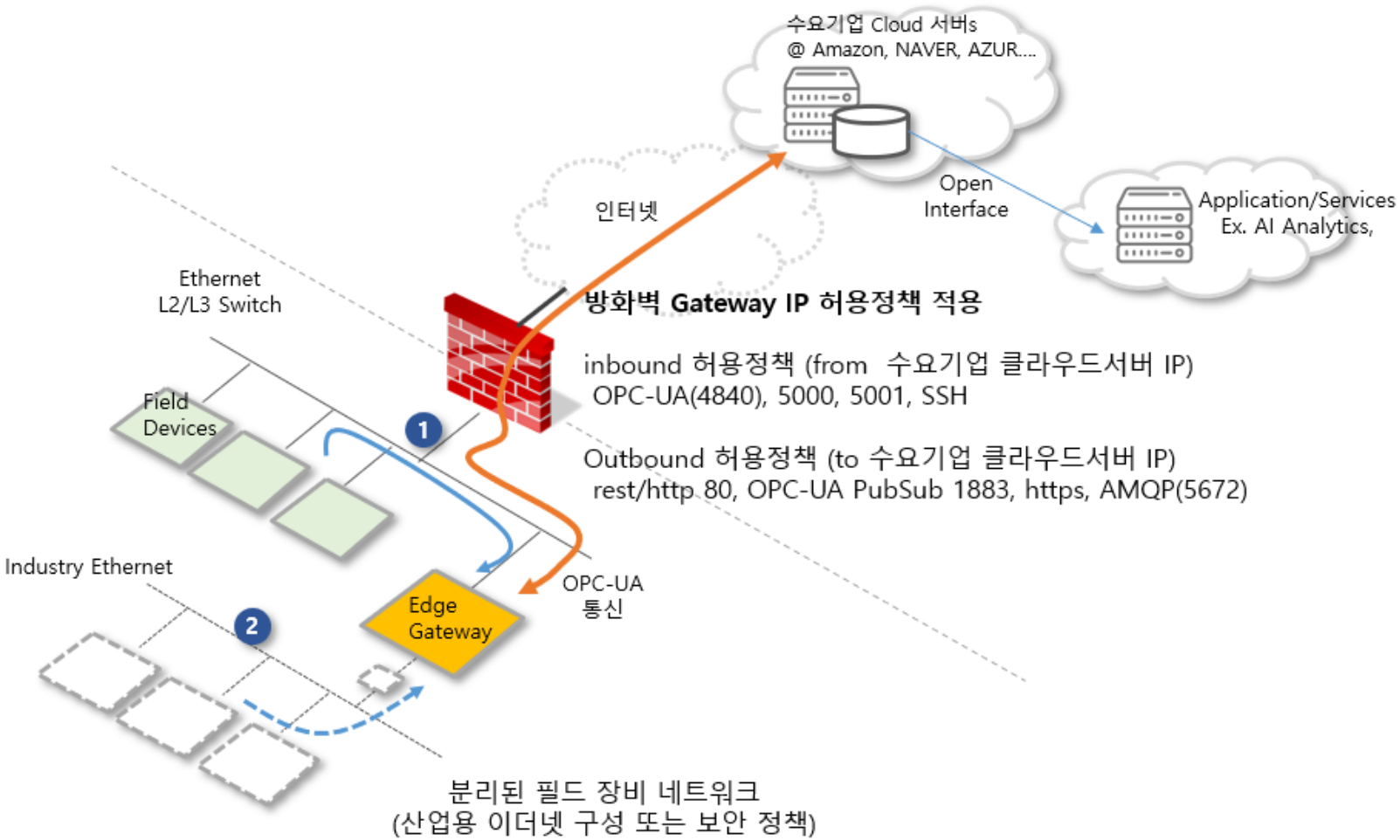
게이트웨이 시스템의 자원을 주기적으로 모니터
(open 소스 tool을 이용)

9 게이트웨이 대시보드

도커/컨테이너가 아닌 호스트 리눅스에서 동작하는 운영어플리케이션: 게이트웨이 대시보드

01. 에지 게이트웨이 소개

◆ 에지 게이트웨이 네트워크 구성 및 통신



01. 에지 게이트웨이 소개

◆ 에지 게이트웨이 시스템 권고 사양

i5 이상의 Multi Core CPU

- Field 장비로부터 실시간 데이터 수집
- 실시간 클라우드 데이터 업로드
- 실시간 로컬 시계열 DB 저장
- 운영 중 데이터 Backup
- Multiple Container 운영
- 향후 에지컴퓨팅 기능 확장

- **50ms~ data interval**
- **수집 Tag 개수**
  ex. 10,000 개 이상의 데이타

OS

- Linux Debian 9
- NOT Support Window

산업용 PC

Multi Network Ports

- 물리적 네트워크 분리용
- (옵션) 산업용 LAN Port 지원

Min. 16 GB DRAM

- 시계열 DB의 메모리 사용
- 멀티플 컨테이너 어플리케이션

Min. 256GB의 SSD Hard disk

- Daily 1GB~ 데이터 저장
- 6개월 데이터 저장
- Backup 용 Disk 용량 여유 확보 (내/외장 hard)

01. 에지 게이트웨이 소개

◆ 에지 게이트웨이 파일럿 프로젝트 예

산업용 PC 하드웨어

Linux OS

Open source (Library, Engine)

컨테이너 Application

Open Source

- Linux Debian 9
- Docker Engine
- Python3 Libraries
- MQTT broker
- Monitoring Util
- OPC UA 애그리게이션 컨테이너
- 시스템 관리 컨테이너

상용

- 시계열DB 컨테이너 (machbase사)

02

시스템 기본 셋업 및 도커 설치

KOREA SMART MANUFACTURING OFFICE

02. 시스템 기본 셋업 및 도커 설치

◆ 리눅스 시스템 기본 셋업-1

리눅스 초기 셋업

- 사용자 계정 새로 생성한다. 생성해야 할 계정은 아래와 같다.

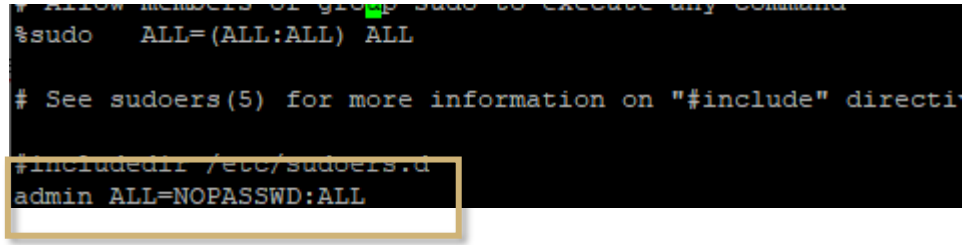
  ID : admin
  PW : nestfield
  Hostname : gateway

- 터미널에서 아래 명령어 입력 시 쉽게 사용자를 추가할 수 있다.
  - admin@gateway:~$ sudo adduser admin

- 그 후 유저 권한을 아래 명령어를 이용해 su로 설정한다.
  - admin@gateway:~$ sudo usermod –aG sudo admin

- 아래 명령어를 입력한다.
  - admin@gateway:~$ sudo visudo

- 맨 아랫줄에 다음 내용을 추가 후 저장한다.
  - admin ALL=NOPASSWD:ALL



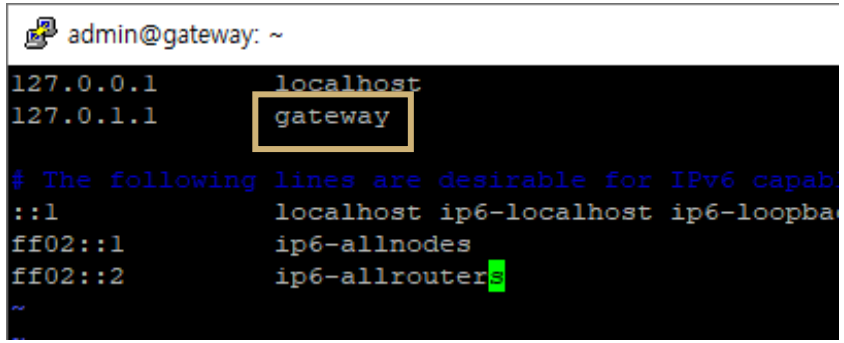
02. 시스템 기본 셋업 및 도커 설치

◆ 리눅스 시스템 기본 셋업-2

리눅스 초기 셋업

- 호스트 이름이 gateway로 설정 되어있지 않다면 아래 명령어를 이용하여 변경한다.
  - ➢ admin@gateway:~$ sudo vi /etc/hostname 입력 후 호스트 이름 설정

- 마찬가지로 아래 명령어를 입력하여 똑같은 호스트 이름으로 설정한다.
  - ➢ admin@gateway:~$ sudo vi /etc/hosts 입력 후 호스트 이름 설정



02. 시스템 기본 셋업 및 도커 설치

◆ 네트워크 설정 및 도커 설치-1

네트워크 셋업 및 도커 설치

- 파일 다운로드 및 통신을 위해 IP주소를 설정한다. 설정 방법은 아래와 같다.
  - ➢ admin@gateway:~$ sudo ifconfig 명령어를 이용해 이더넷 인터페이스의 이름을 확인한다.

```
admin@gateway:~$ sudo vi /etc/network/interfaces
admin@gateway:~$ sudo ifconfig
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.30  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::260:65ff:fe5f:e3af  prefixlen 64  scopeid 0x20<link>
        ether 00:60:65:5f:e3:af  txqueuelen 1000  (Ethernet)
        RX packets 43279815  bytes 6615738380 (6.1 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 74472324  bytes 48205373311 (44.8 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device memory 0xb1400000-b147ffff
```

- 아래 명령어 입력 후 그림과 같이 내용을 추가한다.
  - ➢ admin@gateway:~$ sudo vi /etc/network/interfaces

admin@gateway: ~

```
# interfaces(5) file used by ifup(8) and ifdown(8)
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto enp2s0
iface enp2s0 inet static
address 192.168.2.30
netmask 255.255.255.0
gateway 192.168.2.1
dns-nameservers 8.8.8.8
```

고정IP 사용
사용할 IP 주소
Subnet mask
게이트웨이 주소
DNS 서버 주소

02. 시스템 기본 셋업 및 도커 설치

◆ 네트워크 설정 및 도커 설치-2

네트워크 셋업 및 도커 설치

- IP주소 세팅이 완료되었다면 아래 명령어 중 하나를 입력하거나 시스템을 재시작하여 네트워크 서비스를 재실행시킨다.
  - admin@gateway:~$ sudo service network-manager restart
  - admin@gateway:~$ sudo service networking restart

- 아래 명령어를 이용해 정상적으로 IP주소가 할당되었는지 확인한다.
  - admin@gateway:~$ sudo ifconfig

- 아래 명령어를 이용해 인터넷 연결이 정상적으로 되었는지 확인한다.
  아래 그림과 같이 핑이 제대로 나가면 연결된 것이다.
  - admin@gateway:~$ ping 8.8.8.8

```
admin@gateway:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=114 time=34.128 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=114 time=34.794 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=114 time=34.240 ms
```

- 아래 명령어를 이용해 시스템을 업데이트한다.
  - admin@gateway:~$ sudo apt-get update

- 그 후 아래 명령어를 입력하여 도커를 설치한다.
  - admin@gateway:~$ sudo curl -fsSL https://get.docker.com/ | sudo sh

※ Curl이 설치 되어있지 않다면 아래 명령어로 설치한다.
  - admin@gateway:~$ sudo apt-get install curl

03 에지 게이트웨이 프로그램 설치

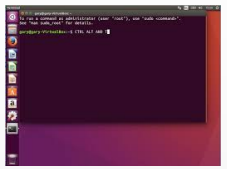
KOREA SMART MANUFACTURING OFFICE

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨용 S/W 설치 및 동작

01

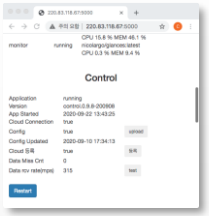
S/W 설치



- Linux (Debian 9 권고) Setup
- Docker pull from docker Hub
  - nestfield/opcua and control
- 컨테이너 구동
- 필수 Python packages 설치
- 기본 config upload
  - 등록 ID, key, doud 주소.

02

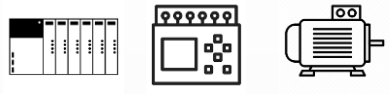
클라우드 등록



- 게이트웨이 관리웹 접속
- 콘트롤 : 클라우드 등록 버튼
- 클라우드 통신 개시
  - AMQP, Restful API
- 클라우드 발급 아이디/키 인증
- Info Model 파일 Download

03

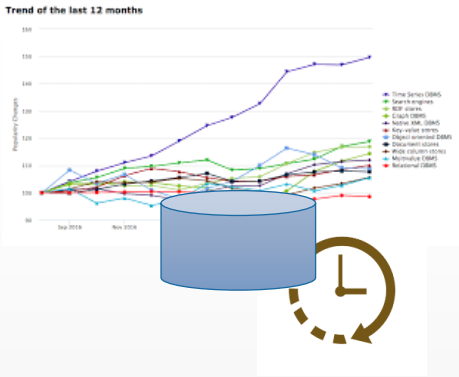
OPC UA 구동



- Aggregation 서버 기동
- - 자동 설정 by Info Model
- Field 장비 접속 및 데이터 수집
- - OPC UA or other
- 클라우드 리포트 이벤트
- - "OPC UA Server Ready"

04

데이터 저장/제공



- 클라우드의 OPCUA 접속
- 실시간 데이터 클라우드 전송
- 실시간 데이터 로컬 저장
- 저장 : 시계열 DB
- - Commercial
- - Open Source
- 데이터 가공 서비스에 데이터 제공

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨용 설치 이미지 및 S/W Repositrory

게이트웨용 S/W 프로그램 또는 이미지 배포

컨테이너
이미지들

nestfield/opcuamodule:relXXXX
nestfield/opcuamodule:relXXXX
machbase/machbase:6.1.X
eclipse-mosquitto:latest
nicolargo/glances

docker HUB

대시보드 프로그램

gatewayHostPackage.tar

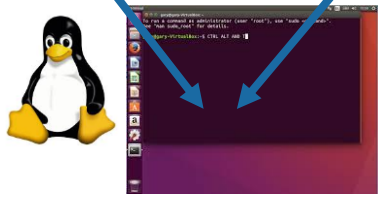
GitHub

Public/Open
Repository

GitLab

Docker pull
명령어로 다운로드

curl download
(url 별도 공지)



03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-1

완료되어야 하는 설치 선행작업

- Docker Engine 설치, 리눅스 부팅 시 docker 자동 실행 확인
- Linux에 admin 사용자 생성, sudo 권한 부여 및 admin 사용자로 login
- 현재 Login한 user(admin)의 Docker 실행 권한 부여
  : `sudo usermod -aG docker $USER`
- **gatewayHostPackage.tar** 다운로드 및 압축 해제
  - tar xvf gatewayHostPackage.tar –C /home/admin/
  - 컨테이너간 공유용 디렉토리 (sharedFolder) 생성 : **/home/admin/sharedFolder**
  - DB Backup용 mount Point 디렉토리 생성 **: /home/admin/exDisk**
- **gateway.config 파일 편집**

설치 및 구동 작업 순서

1	2	3	4
컨테이너 네트워크 생성	도커 어플리케이션 이미지 설치	어플리케이션 컨테이너 구동	컨테이너 구동 후 설정/운영

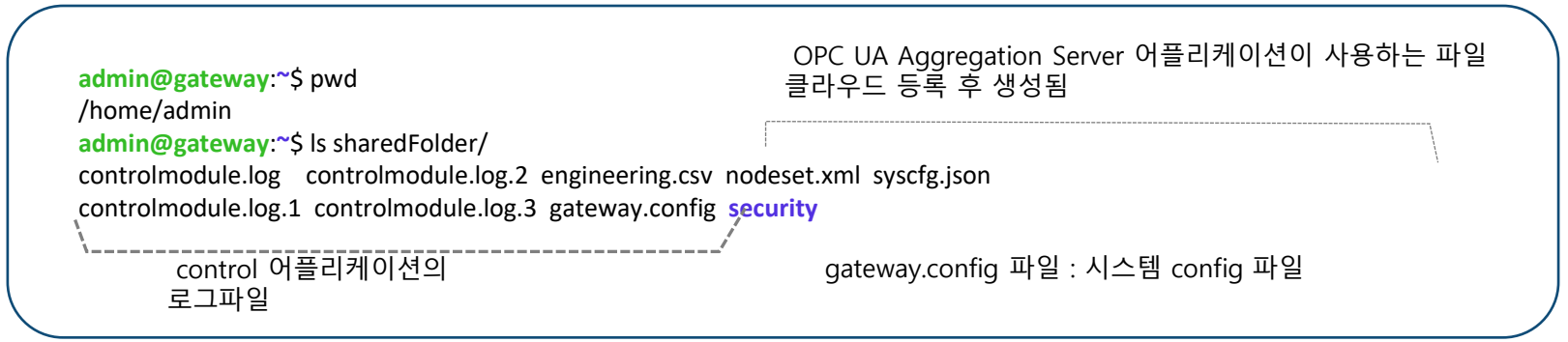
$ docker network create	$ docker pull	$ docker run	$ docker attach
			$ docker restart
			$ docker commit
			$ logroate

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-2

컨테이너 공유볼륨 구조

- Home directory안에 sharedFloder 디렉토리 생성 후 그 하위에 security 디렉토리 추가 생성
- 공유폴더는 모든 어플리케이션들이 접근 가능하도록 각 컨테이너가 구동되며, 각 어플리케이션이 필요한 파일들이 아래 예처럼 자동 생성됨.



- **하위폴더 security** 에는 시스템 또는 각 어플리케이션이 사용할 여러 ID의 암호를 Encrypt하여 저장 → 암호 Encryption 챕터 참조

게이트웨이 config파일 – gateway.config

- 시스템 및 각 어플리케이션이 사용할 설정 사항 및 시스템의 등록, 기동 정보가 업데이트됨
- 게이트웨이 등록과 관련 필수 정보가 운영자에 의해 사전 편집저장되어야 함
  : gatewayName, usbDriveInstalled , gwAdmin, opcuaUser, cloudServer, amqp의 값이 있어야 게이트웨이가 동작 가능

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-3

게이트웨이 config파일 편집

vi 에디터 직접 작성 토는 텍스트로 편집 후 웹으로 업로드

```
admin@gateway:~$ vi sharedFolder/gateway.config
{
  "gatewayName": ”company1-gw",
  ”gatewayShare”:””,
  "hostIP": "",
  "siteFirewall": "",
  "gwAdmin": {
    "id": "admin",
    "credential": "security/admin.secured"
  },
  "opcuaUser": {
    "id": "nestfield",
    "credential": "security/opcua.secured"
  },
  "cloudServer": {
    "ip": ”123.100.200.123",
    "credential": "security/regiKey.secured"
  },
  "amqp": {
    "ip": "123.100.200.123",
    "id": ”amqpID1",
    "credential": "security/amqp.secured"
  },
  "serverRegiCompleted": false,
  "aggServerReadySuccess": false,
  "updated": "2020-09-10 17:34:13"
}
```

게이트웨이를 클라우드에 등록 시 사용되는 게이트웨이 ID
클라우드에 미리 정의되어야 함

Encrypt된 해당 암호의 파일 경로

게이트웨이 대시보드에 로긴하는 ID. (Linux admin 사용자 ID와 다름)

클라우드 OPC UA 클라이언트가 게이트웨이 Aggregation 서버에 접속하기 위한 ID

게이트웨이가 등록할 클라우드 서버와 통신할 AMQP IP Address

클라우드 AMQP 서버 접속을 위한 ID

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-4

Security 디렉토리와 암호 Encryption

- 운영프로그램 dna_encrypt 실행파일을 security 디렉토리 밑에 복사 및 chmod +x
- 아래와 같이 각 암호를 **xxxx.secured** 파일로 Encrypt함 (파일 이름은 gateway.config의 confidential 경로와 동일해야 함.
- 각 암호가 미리 Encrypt되어 파일로 저장되어야 시스템과 어플리케이션이 정상 동작함

게이트웨이의 클라우드 등록 키

```
admin@gateway:~$ ls sharedFolder/security
admin.secured  amqp.secured  dna_encrypt  opcua.secured  regiKey.secured
```

클라우드의 Agg서버 접속 ID의 암호

클라우드 AMQP 접속 ID의 암호

gateway 대시보드 admin 로긴 암호

예 : gateway 대시보드 admin user 암호 Encryption

Encrypt후 저장할 파일명

```
admin@gateway:~/sharedFolder/security$
admin@gateway:~/sharedFolder/security$ ./dna_encrypt admin.secured
enter encryption password: ******
verify encryption password: ******
success encryption!!!
```

Encrypt할 암호

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-5

컨테이너 통신용 도커 내부 네트워크 생성

- 생성 Docker Network Type : bridge
- Network 이름 : "**internalNet**"

관련 명령어

```
$ docker network create --driver bridge internalNet
$ docker network ls
$ docker network inspect internalNet
```

컨테이너 이미지 설치

- 아래 표의 각 container에 해당하는 Image를 docker pull 명령어로 Docker Hub로부터 Download한다
- controlModule/opcuaModule/tsDB용 image는 별도로 권고된 버전을 설치한다.

Container 명	Docker Images
controlModule	nestfield/controlmodule:relXXXX
opcuaModule	nestfield/opcuamodule:relXXXX
tsDB	machbase/machbase:6.1.X
broker	eclipse-mosquitto:latest
monitor	nicolargo/glances

관련 명령어

```
$ docker pull <image_on_docker_hub>

$ docker images
$ docker ps -a
$ docker inspect <container_name>
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-6

어플리케이션 컨테이너 구동

컨테이너 이름	컨네이너 간 통신 시 컨테이너 이름으로 IP Address Resolution하므로 기 정의된 이름 사용해야함 : controlModule, opcuaModule, tsDB, monitor, broker
네트워크	monitor 제외한 나머지 컨테이너가 "internalNet" bridge 네트워크를 사용하며 monitor 컨테이너는 host의 네트워크를 공유함
재기동	Linux host 재부팅 시 또는 container 종료 시 자동으로 컨테이너가 시작하는 옵션 : --restart=always
Port forwarding	컨테이너의 서비스를 Linux의 host의 서비스로 mapping opcuaModule : OPCUA서비스 4840 to host 4840 tsDB: Web 서비스 5001 to host 5001
공유 볼륨	Host에 생성한 sharedFolder 디렉토리를 컨테이너간 공유볼륭(공유 저장소)로 지정 controlModule, opcuaModule

관련 명령어

```
$ docker run -it --name <container이름> --hostname <container호스트명> \
   -p <host_port>:<container_port> \
   -v <공유볼륨이름>:<mount_path> \
   --network <network이름> \
   --restart=always
   <image:tag>
```

예 : docker run

```
$ docker run -it --name opcuaModule --hostname opcuaModule -p 4840:4840
   -v sharedVolume:/project/sharedFolder/ --network internalNet
   --restart=always nestfield/controlmodule:rel0831
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-7

어플리케이션 컨테이너 구동 확인

- docker ps 명령으로 run 명령과 옵션을 통해 5개의 코어 어플리케이션 컨테이너들이 정상 기동이 되었는지 확인

관련 명령어

```
$ docker ps
$ docker images
$ docker stats
```

예 : docker ps

```
admin@gateway:~$ docker ps
CONTAINER ID  IMAGE                    COMMAND             CREATED     STATUS    PORTS                    NAMES
582599b3de7f  nicolargo/glances        "/bin/sh -c 'python3…"  5 hours ago  Up 3 mins                      monitor
905d7d77c27d  nestfield/opcuamodule:rel0828  "/bin/bash"          5 hours ago  Up 3 mins 0.0.0.0:4840->4840/tcp  opcuaModule
c4cb2148f0dd  machbase/machbase:6.1.8  "/bin/sh -c 'bash -C…"  5 hours ago  Up 3 mins 5656/tcp, 0.0.0.0:5001->5001/tcp  tsDB
11437c01915e  eclipse-mosquitto:latest  "/docker-entrypoint.…"  5 hours ago  Up 3 mins 1883/tcp               broker
d9eb254fc74e  nestfield/controlmodule:rel0828 "/bin/bash"          5 hours ago  Up 3 mins                      controlModule
```

예 : docker stats

```
admin@gateway:~$ docker stats --no-stream
CONTAINER ID    NAME           CPU %     MEM USAGE / LIMIT    MEM %     NET I/O          BLOCK I/O        PIDS
582599b3de7f    monitor        0.26%     272.4MiB / 14.96GiB  1.78%     0B / 0B          48.8MB / 0B      7
905d7d77c27d    opcuaModule    26.84%    150.1MiB / 14.96GiB  0.98%     64.3GB / 227GB   591MB / 2.73GB   38
c4cb2148f0dd    tsDB           15.44%    6.66GiB / 14.96GiB   44.52%    10.9GB / 429MB   476MB / 4.41GB   56
11437c01915e    broker         2.90%     921.2MiB / 14.96GiB  6.01%     199GB / 197GB    3.81MB / 0B      1
d9eb254fc74e    controlModule  8.42%     22.45MiB / 14.96GiB  0.15%     130GB / 104GB    154MB / 29.6MB   8
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-8

컨테이너 구동 후 설정/작업 사항

- 시계열 DB(tsDB)의 tag 테이블 생성
- 시계열 DB(tsDB)의 라이센스 적용
- 시계열 DB(tsDB) max 메모리 usage 수정 → 첨부 참조
- 시계열 DB(tsDB) MWA 로깅 파일 사이즈 제한 (중요) → 첨부 참조
- Docker 컨테이너 로깅 설정
- 컨테이너 이미지 백업

동작중인 컨테이너 접속

- Attach 또는 exec 명령으로 동작중인 컨테이너에 접속
- Attach 명령으로 접속 후 빠져나올 때 반드시 <ctrl> + p 와 <ctrl> + q를 사용하고 exit을 사용하지 말아야 함.
- Exec 명령은 exit로 logout 가능

관련 명령어

```
$ docker attach <컨테이너 이름>
$ docker exec -it <컨테이너 이름> bash
```

예 : docker attach

admin@gateway:~$ docker attach tsDB
machbase@tsDB:~$

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-9

컨테이너 구동 후 작업 – tsDB의 tag 테이블 생성

- 컨테이너 tsDB에 접속해서 machsql를 실행 (password : manager)
- Tag table 생성 (컬럼 : name, time, value)
- (** table생성 60초정도 소요)

관련 명령어

```
$ machbase@tsDB:~$ machsql

$ Mach> create tagdata table tag(name varchar(256) primary key, time datetime
basetime, value double summarized);      # tag table 생성 명령
$ Mach> desc tag                        # 생성된 table 확인
```

예 : desc tag

```
Mach> desc tag;
[ COLUMN ]
-------------------------------------------------------------------------------------------------
NAME                                    NULL?     TYPE              LENGTH
-------------------------------------------------------------------------------------------------
NAME                                    NOT NULL varchar            256
TIME                                    NOT NULL datetime           31
VALUE                                   NOT NULL double             17
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-10

컨테이너 구동 후 작업 – tsDB의 라이센스 적용

- 컨테이너 tsDB에 접속해서 machhostid 명령어로 host ID를 확인하여 마크베이스사에 라이센스파일 요청
- 라이센스 파일을 **$MACHBASE_HOME/conf** 디렉토리에 **license.dat** 로 저장
- 마크베이스 데몬 재실행 (machadmin 명령어)

관련 명령어

```
$ machbase@tsDB:~$ machhostid                # 호스트 ID 추출

$ machbase@tsDB:~$ machadmin -s              # 마크베이스 데몬 종료
$ machbase@tsDB:~$ machadmin -f               # 라이센스 확인
$ machbase@tsDB:~$ machadmin -u               # 마크베이스 데몬 시작
```

예 : machadmin -f

```
machbase@tsDB:~$ machadmin -f
------------------------------------------------------------------
   Machbase Administration Tool
   Release Version - 6.1.8.official
   Copyright 2014, MACHBASE Corp. or its subsidiaries
   All Rights Reserved
------------------------------------------------------------------
              INFORMATION
Install Date              : 2020-10-13 14:29:56
Company#ID-ProjectName    : NESTFIELD#0-CLOUDPROJECT
License Policy            : SIZE4DAY
License Type(Version 2)   : OFFICIAL
Host ID                   : 0242AC1200055452
Issue Date                : 2020-08-01
Expiry Date               : None-
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-11

시계열 DB MWA서버 로깅제한

- MWA는 마크베이스사 시계열DB의 구성요소로서 DB SQL의 restful API 지원을 포함
- 컨테이너 tsDB : /home/machbase/machbase/webadmin/flask/MWA.conf의 MWA_LOG_WRITE_REQUEST_PARAMETER 옵션 변경 (= N)

관련 명령어

```
admin@gateway:~$ docker exec -it tsDB bash              # 시계열 DB 접속
machbase@tsDB:~/machbase/webAdmin/flask$ vi MWA.conf    # MWA.conf 편집
machbase@tsDB:~/machbase/bin$ ./MWAserver restart       # MWA 서버 재시작
```

```
##############################################################################
# If this value set to 'Y', INFO level log is recorded. (Y/N)
##############################################################################
MWA_LOG_WRITE_INFO = Y
SCHEDULE_LOG_WRITE_INFO = Y
MWA_LOG_WRITE_REQUEST_PARAMETER = N
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-12

컨테이너 구동 후 작업 – 도커 컨테이너 로깅(logrotate) 설정

- 각 컨테이너의 로그가 과도하게 쌓이는 것을 방지하기 위해 log 파일의 rotation 옵션을 조정
- /var/lib/docker/containers/ 에 컨테이너별 로그 저장
- 도커 로그는 Daily cron job으로 로그를 관리하도록 기설정되어 있으므로 logrotate 옵션만을 조정
- 7일간의 log를 유지하고 파일 Size는 최대 1M로 유지

예 : logrotate 옵션 파일 편집 및 적용

```
admin@gateway:~$ sudo vi /etc/logrotate.d/docker-container    # 옵션 파일 편집

/var/lib/docker/containers/*/*.log {
rotate 7
daily
compress
size=1M
missingok
delaycompress
copytruncate


}
admin@gateway:~$ sudo logrotate /etc/logrotate.conf          # 설정된 로그 옵션 파일 적용
```

03. 에지 게이트웨이 프로그램 설치

◆ 코어 어플리케이션 설치 및 구동-13

컨테이너 구동 후 작업 – 콘트롤 모듈 로그파일 설정(logrotation)

- 콘트롤 모듈 프로그램 로그가 과도하게 쌓이는 것을 방지하기 위해 log 파일의 rotation 옵션을 조정
- /home/admin/sharedFolder 디렉토리에 controlmodule.log가 생성되고
- 이전 3일간의 로그가 controlmodule.log.1, controlmodule.log.2, controlmodule.log.3으로 순환 관리됨
- 3일간의 log를 유지하고 파일 Size는 최대 100K로 유지

예 : logrotate 옵션 파일 편집 및 적용

```
admin@gateway:~$ sudo vi /etc/logrotate.d/controlModuleLog      # 옵션 파일 편집

/home/admin/sharedFolder/controlmodule.log {
 daily
 size 100k
 create 644 admin admin
 rotate 3
 copytruncate
}
admin@gateway:~$ sudo logrotate /etc/logrotate.conf             # 설정된 로그 옵션 파일 적용
```

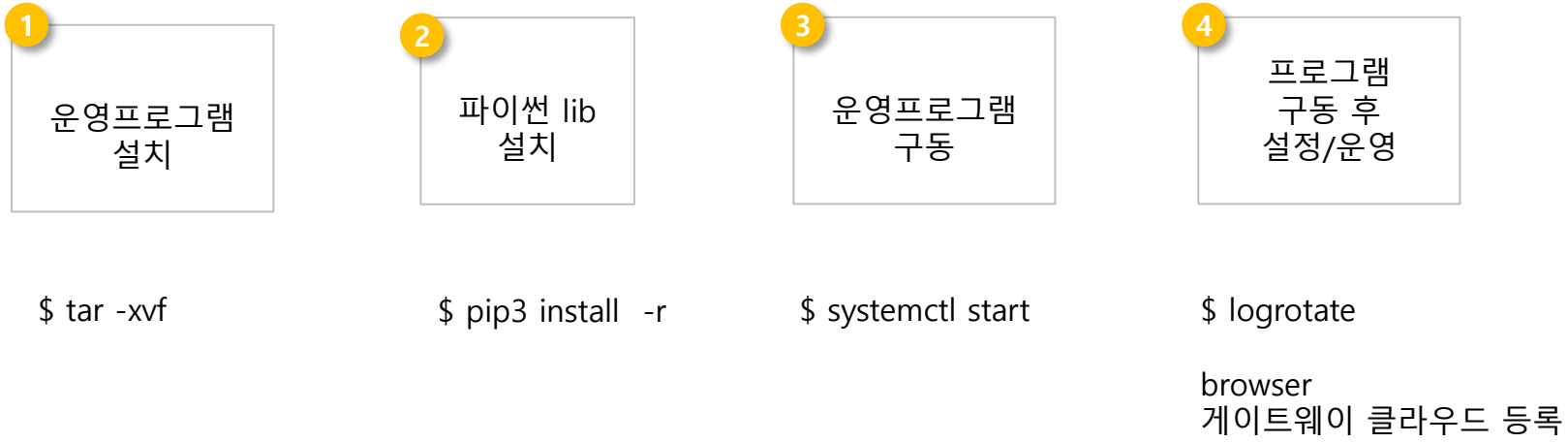
03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-1

완료되어야 하는 설치 선행작업

- 코어 어플리케이션 설치 및 구동
- Python3, PIP3 설치
- /home/admin에 projects 디렉토리 생성
- 운영프로그램 패키지 **gatewayHostPackage.tar** 다운로드

설치 및 구동 작업 순서



03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-2

운영프로그램 패키지 설치

- 리눅스 호스트 설치 프로그램. (컨테이너 프로그램 아님)
- gatewayWebPackage.tar 패키지 압축 projects 디렉토리에 압축풀기 (tar –xvf 명령어)

압축해제 후 파일

```
admin@nestfield:~/projects$ ll
total 60
-rw-r--r-- 1 admin admin  1912 Aug 30 12:43 DockerHandler.py
-rw-r--r-- 1 admin admin  3229 Aug 30 12:43 FlaskAppWrapper.py
-rw-r--r-- 1 admin admin  2687 Aug 30 12:43 GwConfig.py
-rw-r--r-- 1 admin admin  4503 Aug 30 12:43 MqttHandler.py
drwxr-xr-x 2 admin admin  4096 Aug 30 12:45 common
-rw-r--r-- 1 admin admin 13934 Aug 30 12:43 gatewayWeb.py
-rw-r--r-- 1 admin admin   126 Aug 30 12:44 requirements.txt
-rwxr-xr-x 1 admin admin   206 Aug 30 12:44 run.sh
-rwxr-xr-x 1 admin admin    40 Aug 30 12:44 startGatewayWeb.sh
-rwxr-xr-x 1 admin admin    72 Aug 30 12:44 stopGatewayWeb.sh
drwxr-xr-x 2 admin admin  4096 Aug 30 12:45 templates
```

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-3

파이썬 Lib설치

- 패키지 압축 해제 후 requirements.txt를 이용한 패키지 설치

관련 명령어

```
admin@gateway:~/projects$ pip3 install -r requirements.txt
```

설치 lib (requirements.txt)

```
docker==4.2.2
Flask==1.1.2
Flask-Bootstrap==3.3.7.1
Flask-Cors==3.0.8
Flask-SocketIO==4.3.2
paho-mqtt==1.5.0
requests==2.24.0
schedule==0.6.0
Werkzeug==1.0.1
```

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-4

운영프로그램 구동 후 설정/운영 – 시스템 서비스 등록

- 시스템 서비스로서 등록 하고
- 시스템 부팅시 자동 실행(default service) 하도록 설정

관련 명령어

```
admin@gateway:~/projects$ sudo vi /etc/systemd/system/gatewayWeb.service
                                                    # 시스템 서비스 등록 편집
admin@gateway:~/projects$ sudo systemctl daemon-reload
admin@gateway:~/projects$ sudo systemctl enable gatewayWeb.service
```

```
admin@gateway:~/projects$ sudo vi /etc/systemd/system/gatewayWeb.service
[Unit]
Description=gatewey Web interface

[Service]
User=admin
ExecStart=/bin/bash /home/admin/projects/run.sh
WorkingDirectory=/home/admin/projects/

[Install]
WantedBy=multi-user.target
```

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-5

운영프로그램 구동

- Python3 에 의해 구동되는 Flask 웹 어플리케이션으로서
- 웹 어플리케이션 gatewayWeb을 backend로 구동
- 동일 디렉토리에 gatewayWeb.log 로그 파일 생성

관련 명령어

```
admin@gateway:~/projects$ sudo systemctl start gatewayWeb.service # 시스템 서비스 구동 명령어
admin@gateway:~/projects$ sudo systemctl stop gatewayWeb.service  # 시스템 서비스  종료 명령어
admin@gateway:~/projects$ sudo systemctl status gatewayWeb.service  # 시스템 서비스 구동 확인
```

예 : systemctl status 를 통한 서비스 확인

```
admin@nestfield:~/projects$ systemctl status gatewayWeb.service
● gatewayWeb.service - gatewey Web interface
   Loaded: loaded (/etc/systemd/system/gatewayWeb.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2020-10-25 23:08:31 KST; 10min ago
 Main PID: 2675 (bash)
    Tasks: 5 (limit: 4915)
   Memory: 97.7M
      CPU: 4.175s
   CGroup: /system.slice/gatewayWeb.service
           ├─2675 /bin/bash /home/admin/projects/run.sh
           └─2676 python3 gatewayWeb.py
admin@nestfield:~/projects$
```

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-6

운영프로그램 구동 후 설정/운영

- 로그 파일 로그로테이션 설정
- Linux 시스템 부팅 시 자동 시작 설정 (시스템 서비스 등록)
- Optional : 방화벽 포트 open

03. 에지 게이트웨이 프로그램 설치

◆ 게이트웨이 대시보드 설치-7

운영프로그램 구동 후 설정/운영 – 로그파일 설정(logrotation)

- 운영프로그램 로그가 과도하게 쌓이는 것을 방지하기 위해 log 파일의 rotation 옵션을 조정
- /home/admin/projects 디렉토리에 gatewayWeb.log가 생성되고
- 이전 3일간의 로그가 gatewayWeb.log.1, gatewayWeb.log.2, gatewayWeb.log.3으로 순환 관리됨
- 3일간의 log를 유지하고 파일 Size는 최대 100K로 유지

예 : logrotate 옵션 파일 편집 및 적용

```
admin@nestfield:~/projects/package/temp$ vi /etc/logrotate.d/gatewayWebLog
/home/admin/projects/gatewayWeb.log {
 daily
 size 100k
 create 644 admin admin
 rotate 3
 copytruncate
}
```

04

에지 게이트웨이 운영

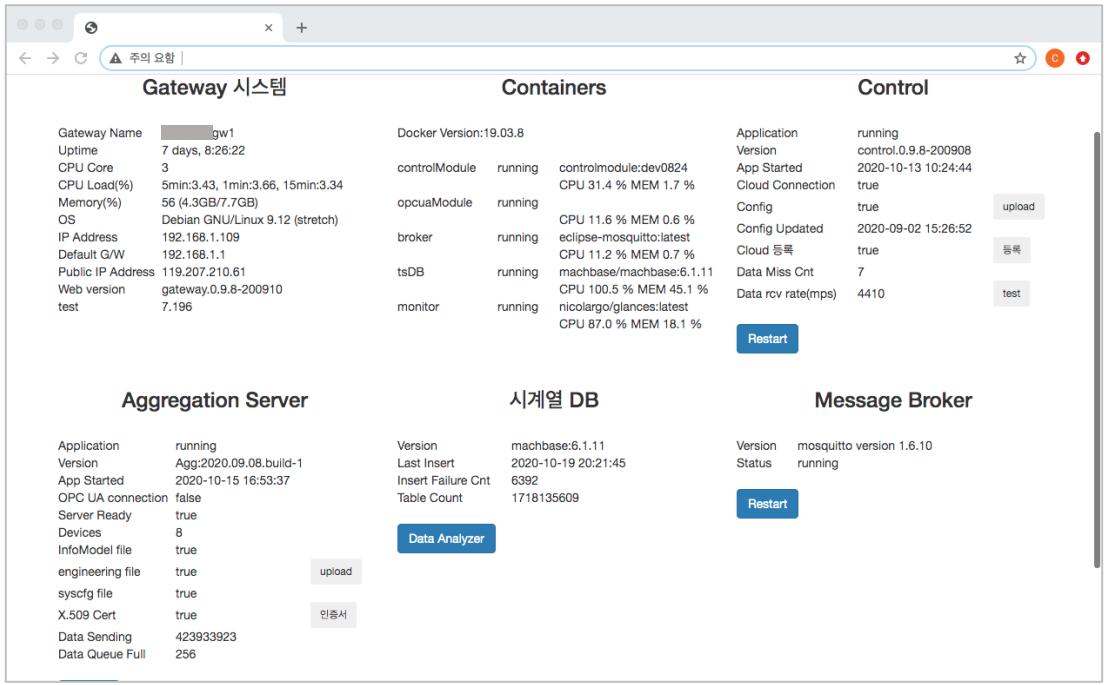
KOREA SMART MANUFACTURING OFFICE

04. 에지 게이트웨이 운영

◆ 게이트웨이 대시보드 기능-1

운영프로그램 - 대시보드

- 일반공급기업의 시스템 관리운영자 또는 공급기업이 사용
- http://<ip_addreess>:5000으로 접속
- 초기 로긴 ID 및 password : admin/nestfield
  (암호 변경은 gateway.config의 gwAdmin에 대한 암호 설정 변경)
- 에지게이트웨이 시스템의 동작 모니터링 및 클라우드 등록

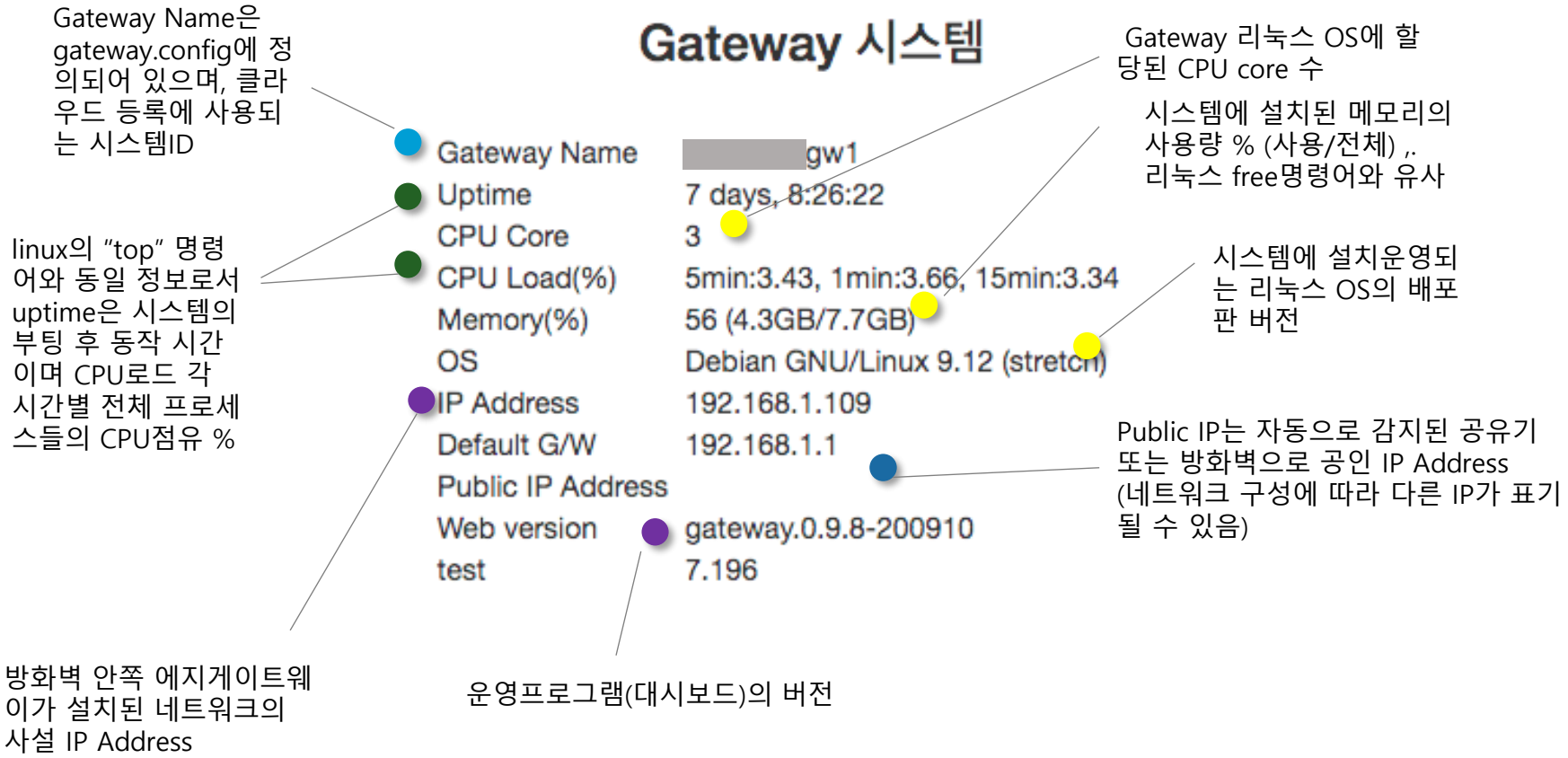


04. 에지 게이트웨이 운영

◆ 게이트웨이 대시보드 기능-2

대시보드 – Gateway 시스템

- 에지 게이트웨이 시스템의 현재 자원 및 운영 시스템 정보



04. 에지 게이트웨이 운영

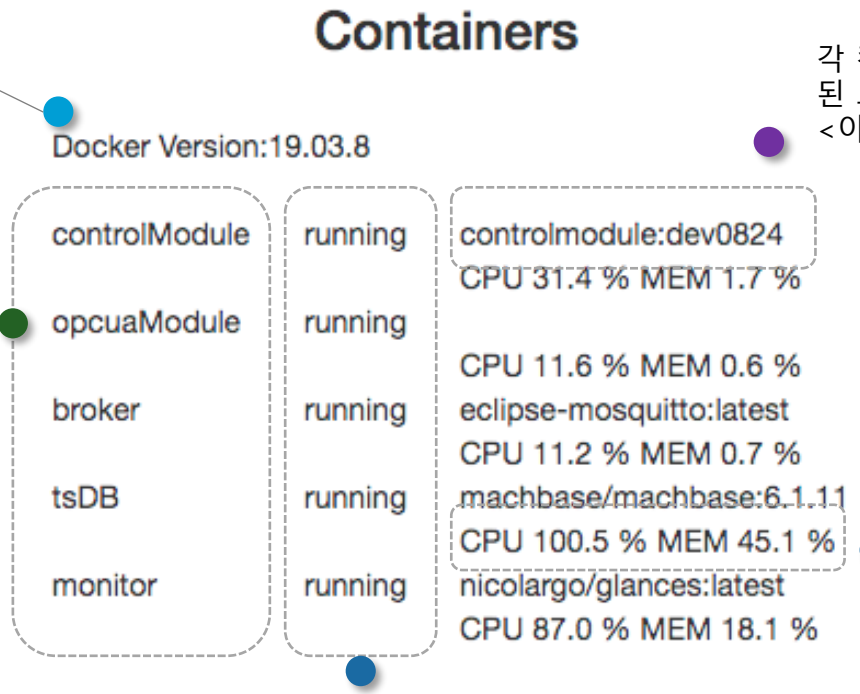
◆ 게이트웨이 대시보드 기능-3

대시보드 – Containers

- 에지 게이트웨이에서 동작하는 도커 컨테이너들의 동작 상태와 자원 점유 정보

현재 게이트웨이 시스템에 설치 운영되고 있는 Docker Engine의 버전정보

현재 게이트웨이 시스템에 설치 운영되고 어플리케이션 컨테이너들의 이름



각 컨테이너 구동에 사용된 도커이미지 <이미지이름>:<버전>

각 컨테이너별 CPU 및 메모리 점유율 docker stats명령어 정보와 유사

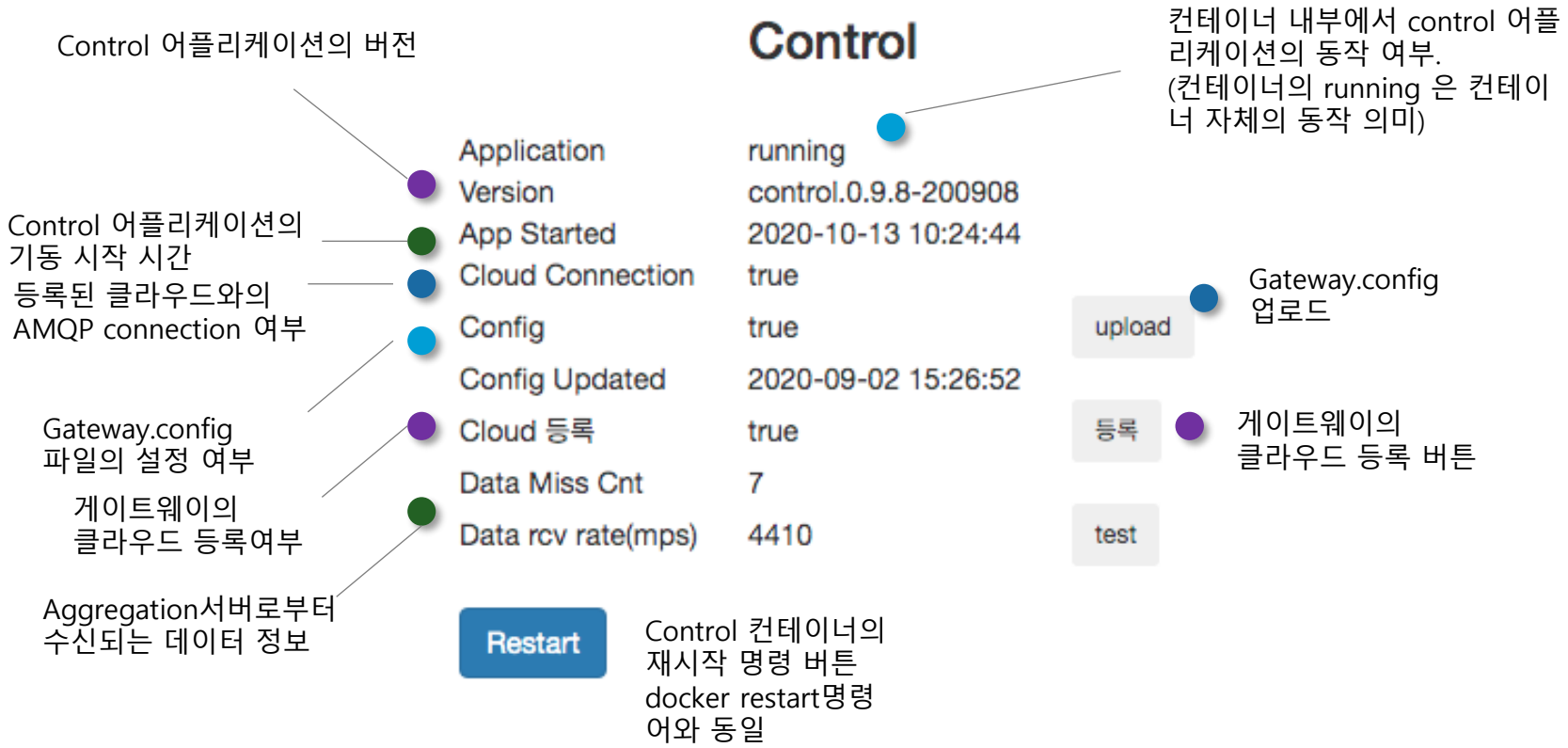
어플리케이션 컨테이너들의 현재 상태 docker ps 명령의 status와 동일

04. 에지 게이트웨이 운영

◆ 게이트웨이 대시보드 기능-4

대시보드 – Control

- control 어플리케이션 컨테이너의 동작 및 에지게이트웨이의 클라우드 등록 기능

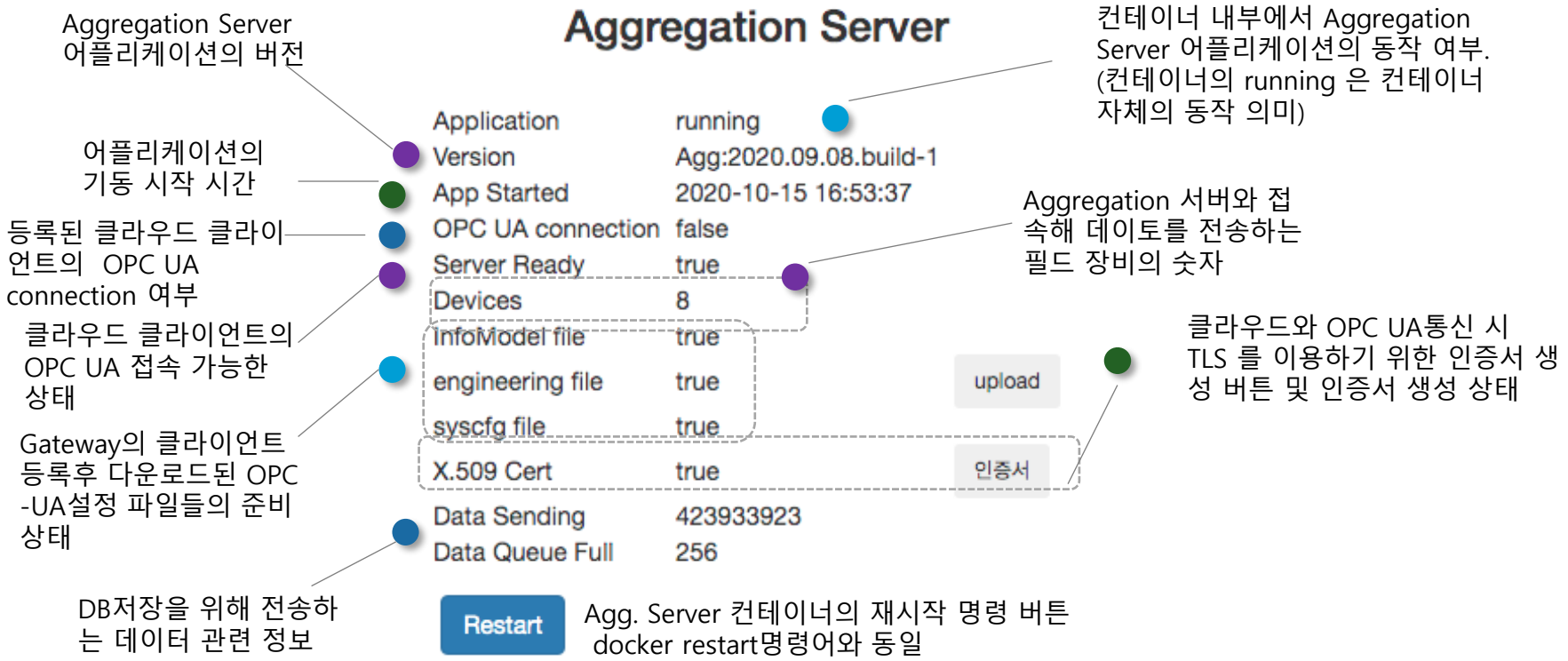


04. 에지 게이트웨이 운영

◆ 게이트웨이 대시보드 기능-5

대시보드 – Aggregation Server

- OPCUA Aggregation 컨테이너의 동작 및 클라우드 통신용 인증서 생성

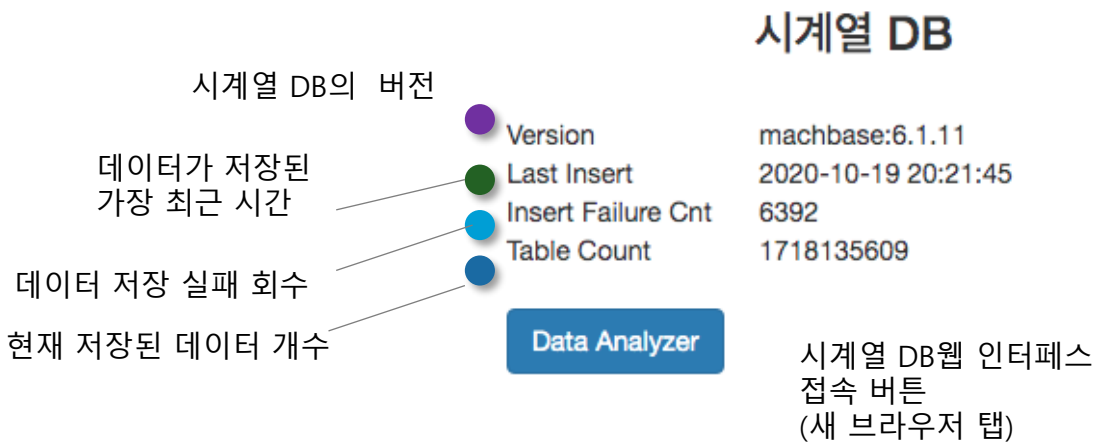


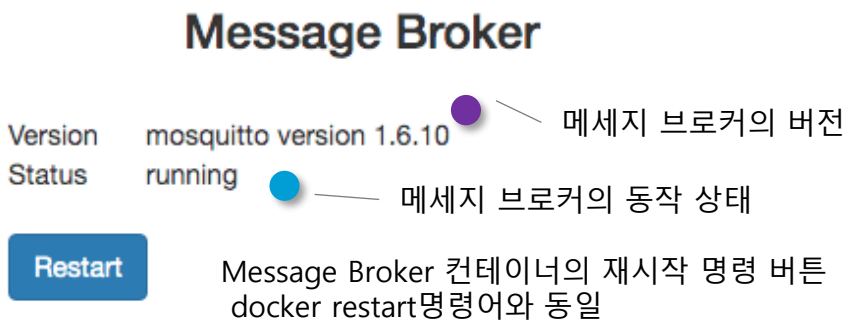
04. 에지 게이트웨이 운영

◆ 게이트웨이 대시보드 기능-6

대시보드 – 시계열 DB, Message Broker

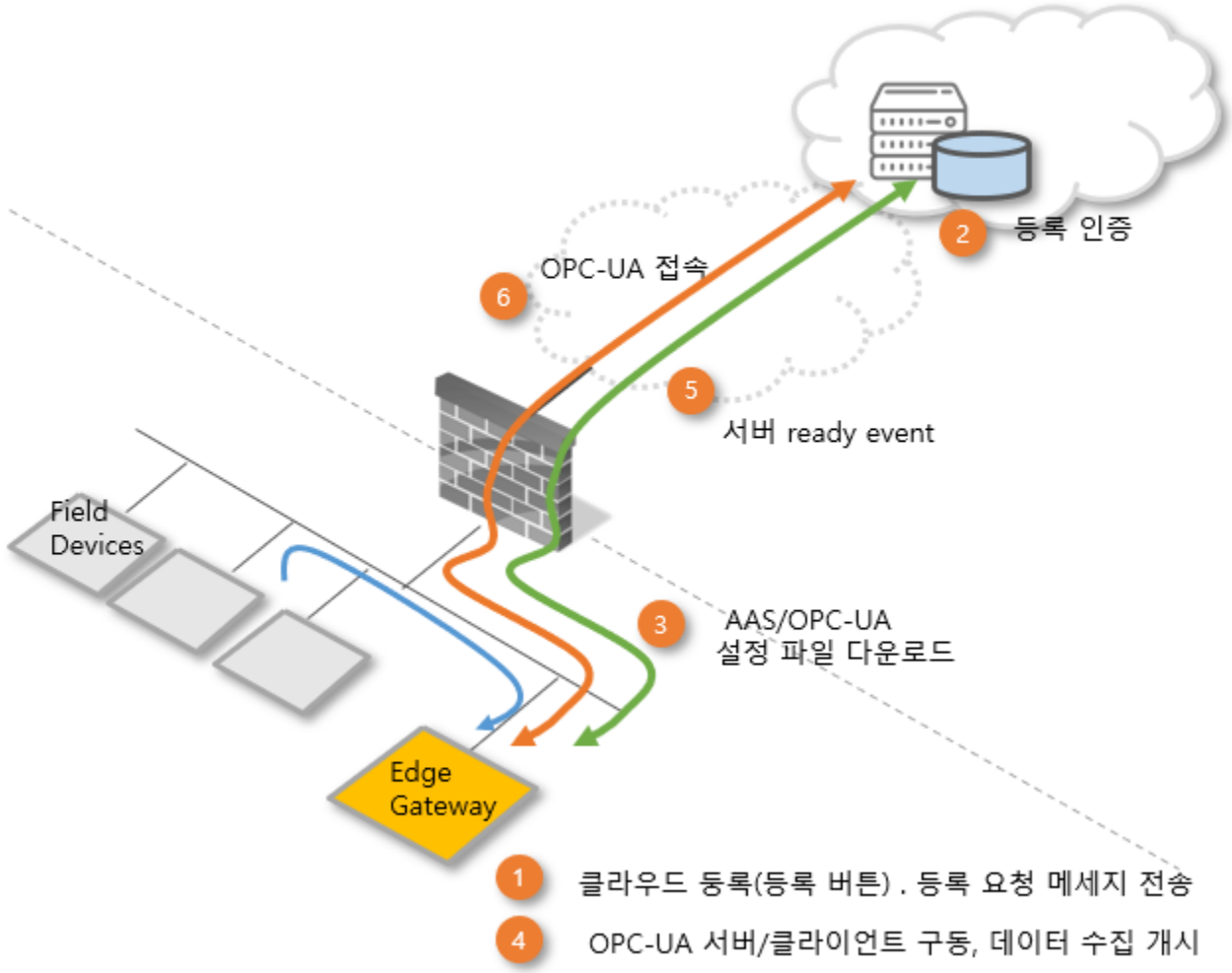
- Tag데이터를 저장하는 시계열 DB 컨테이너의 동작 및 시스템 내부 통신 브로커의 동작 상태





04. 에지 게이트웨이 운영

◆ 게이트웨이 등록 및 클라우드 opc ua 통신 개시-1

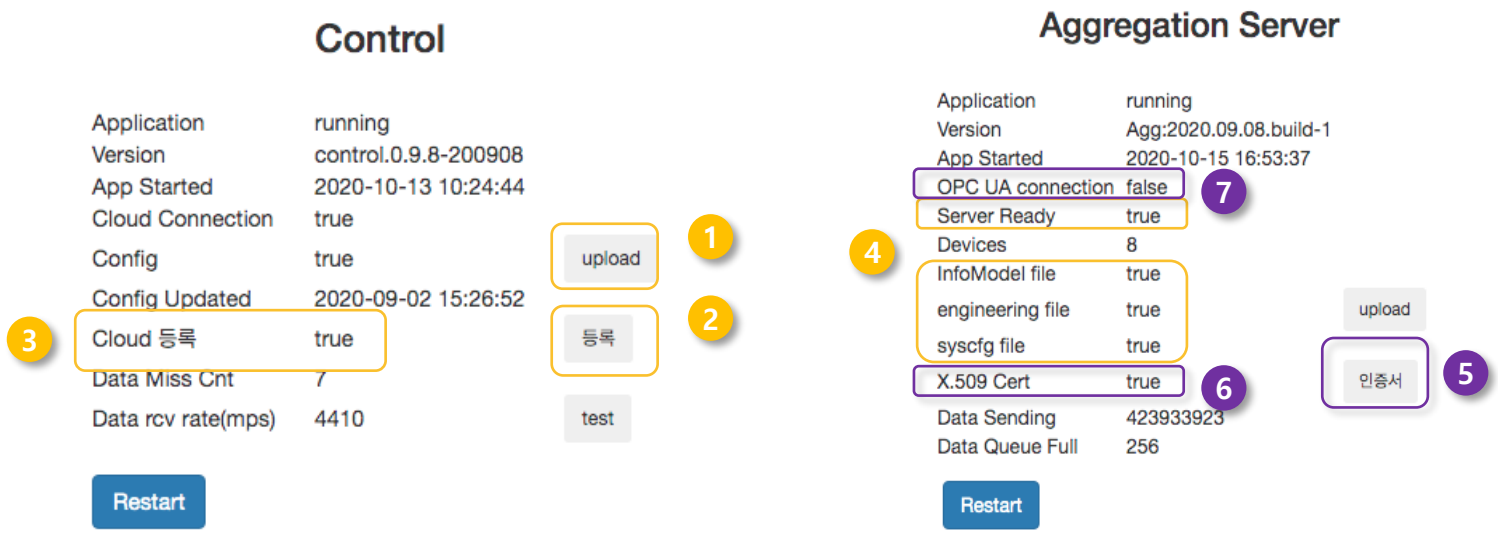


04. 에지 게이트웨이 운영

◆ 게이트웨이 등록 및 클라우드 opc ua 통신 개시-2

게이트웨이 등록 및 최초 가동

- 클라우드 시스템에 등록된 데이트웨이 ID와 기타 접속/암호를 입수하여 gateway.config를 편집/업로드 후(1)
- 게이트웨이 대시보드 – contro의I 등록 버튼(2)으로 눌르고 아래 번호 순서대로 값의 변화를 확인

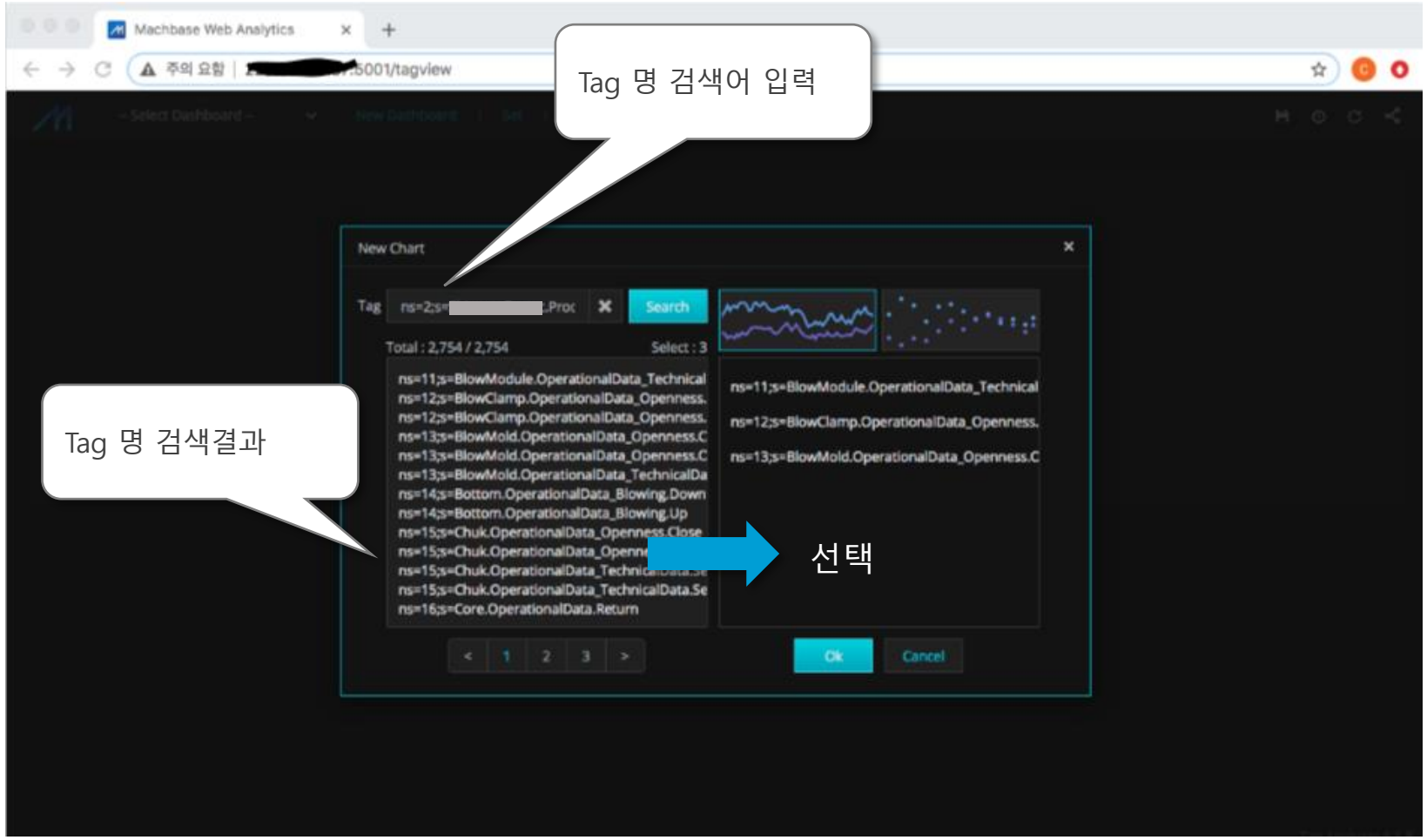


- 1~4까지 등록절차가 확인되면 Aggregation Server의 인증서 버튼(5)을 누르고 인증서 생성을 확인(6)하고
- OPC UA connection 이 확인(7)되면 데이터의 클라우드전송 절차가 개시된 것임

04. 에지 게이트웨이 운영

◆ 로컬 Tag 데이터뷰-1

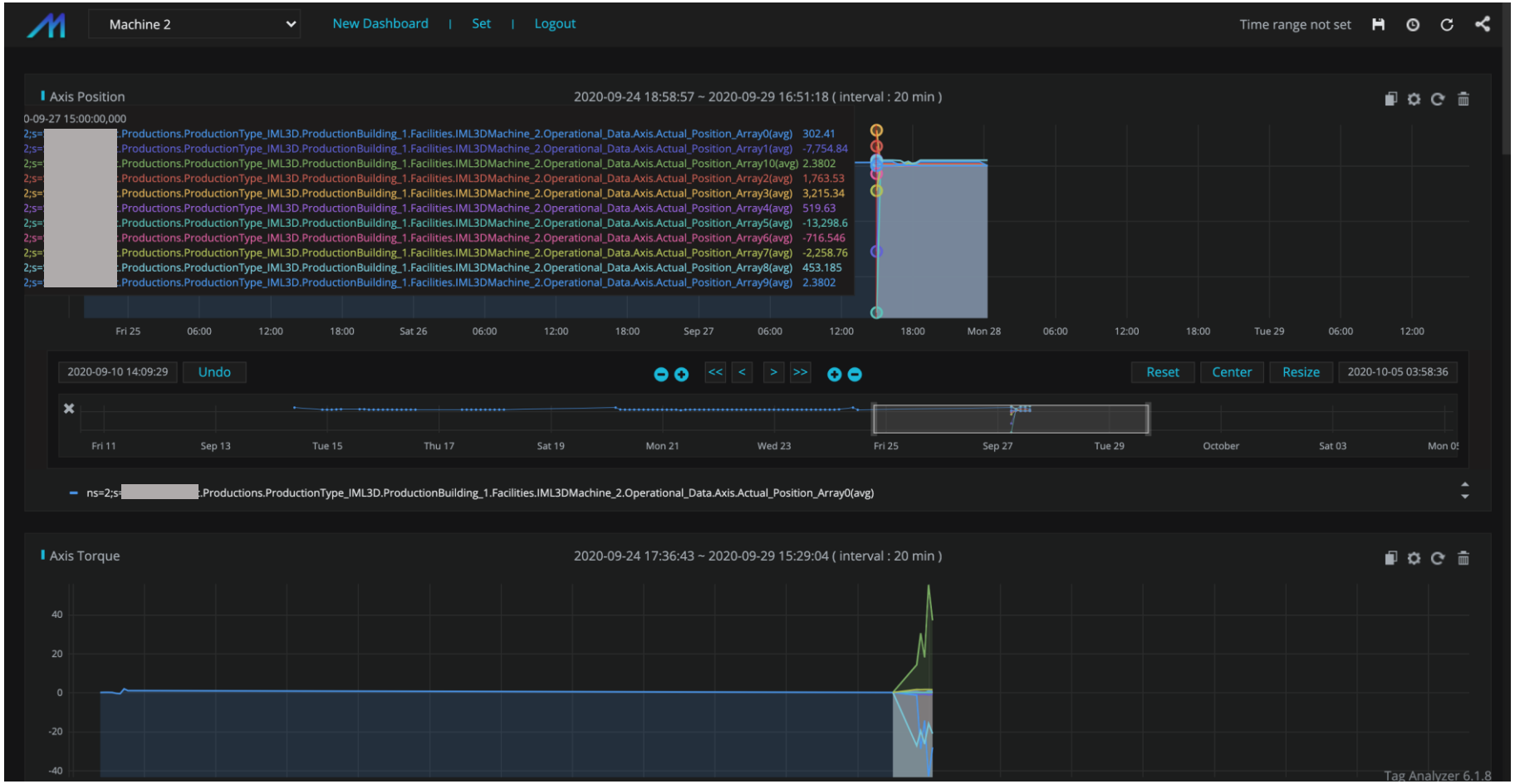
컨테이너 구동 후 작업 – tsDB 데이터의 tag view analyzer 설정



04. 에지 게이트웨이 운영

- 로컬 Tag 데이터뷰-2

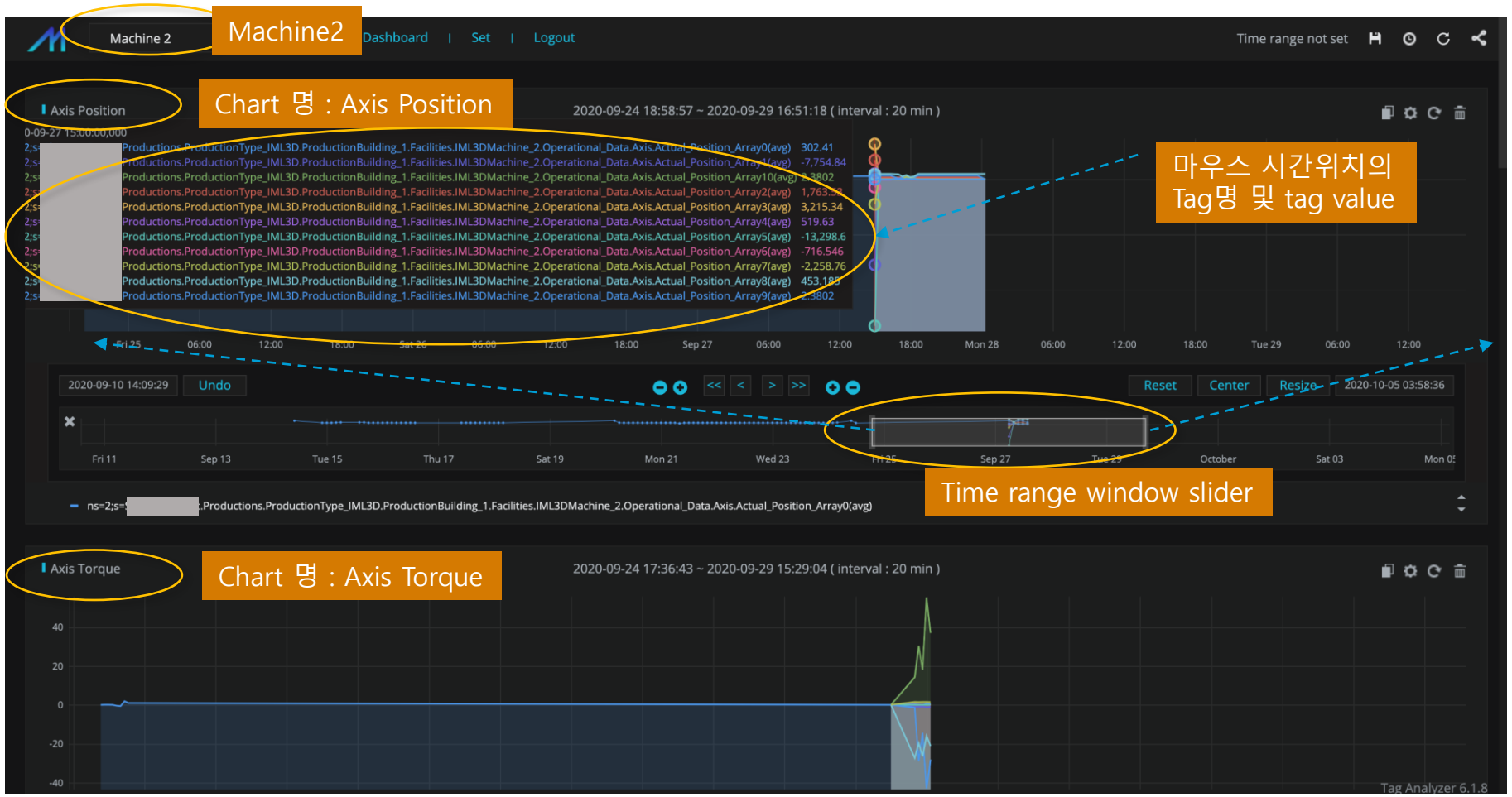
컨테이너 구동 후 작업 – tsDB 데이터의 tag view analyzer 설정



04. 에지 게이트웨이 운영

◆ 로컬 Tag 데이터뷰-3

컨테이너 구동 후 작업 – tsDB 데이터의 tag view analyzer 설정

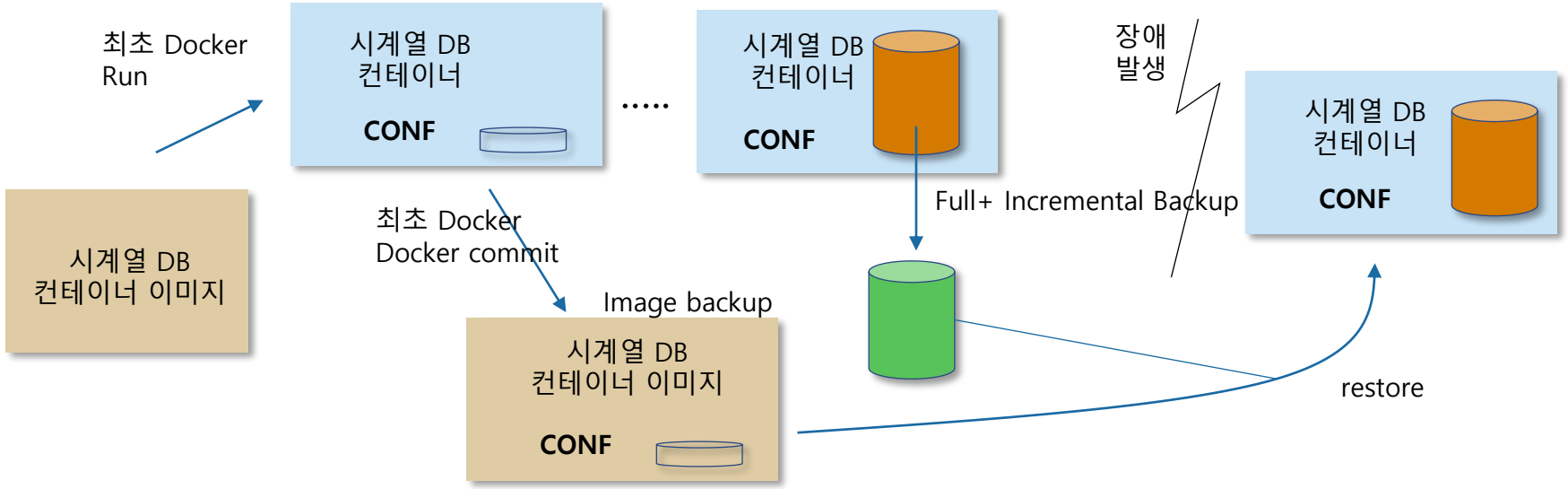


04. 에지 게이트웨이 운영

◆ 시계열 DB 백업 및 Restore

시계열 DB의 로컬 백업

- 디스크용량의 확보 시 클라우드 백업 외 추가 로컬 백업 가능
- 로컬 백업 시 외장하드 권고
- 일정 주기에 Full backup 후 추가적으로 Incremental Backup수행
  ex. 매주 월요일 새벽에 Full backup, 평일 새벽 incremental Backup
- 필요시 비정기 수동 백업 (machsql 명령 사용)
- 최초 시계열 DB 컨테이터 기동 후 컨테이너 image backup



04. 에지 게이트웨이 운영

◆ 컨테이너 이미지 백업

컨테이너 구동 후 작업 – 도커 컨테이너 이미지 백업

- 장애 시 손쉬운 복구를 위해서 컨테이너의 최초 정상 구동 및 운영 개시 후 각 컨테이너를 커미트하여 이미지로 저장. (docker commit, save 명령)
- 현재 운영컨테이너의 삭제 후 기존 백업 컨테이너 이미지로 복구 (docker load, run 명령)

관련 명령어

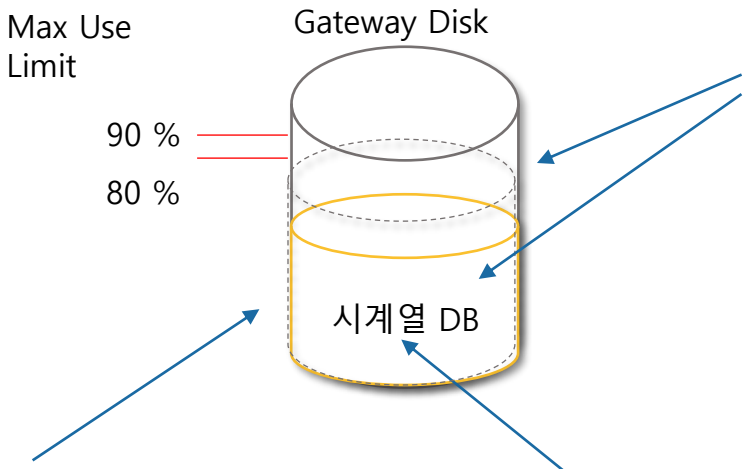
```
$ docker commit <container_name> <imageName>:<tag>      # 현재 컨테이너를 이미지로 커미트
$ docker save -o <compress_name>.tar <imageName>:<tag> # 이미지를 압축

$ docker load -i <compress_name>.tar  # image를 동일 호스트에 복구 또는 다른 호스트에 로딩
```

예 : tsDB 컨테이너 이미지 백업

04. 에지 게이트웨이 운영

◆ 디스크 용량 관리-1

하드디스크 사용 및 시계열 DB 데이터 삭제



IPC 전체 Disk usage 와 시계열 DB 데이터 증가량을 1 분 간격으로 측정

IPC 전체 Disk usage 가 Max Use를 넘으면 시계열 DB에서 가장 old 데이터를 Disk Max Use단계별(80%, 90%)로 차별적으로 삭제.

최초 Disk 사용량 : 약 10G
(현재 구현 설치된 프로그램 및 컨테이너)

Disk 증가분 : DB +DB backup + Logs

설정된 데이터 Tag의 sampling 간격과 값변화에 따라
저장 데이터량과 저장기간 달라짐
cf. 마크베이스 시계열 DB기준 최초 DB Disk사용량(데이터 수집 전) : 약 3G

04. 에지 게이트웨이 운영

◆ 디스크 용량 관리-2

컨테이너 구동 후 작업 – 하드디스크 용량 확인 및 시계열 DB 데이터 삭제

- 시계열 DB 컨테이너의 정기적 이미지 백업과 별도로
- 정기적으로 하드디스크의 용량을 확인(df -h)하고, 디스크 용량 부족 시 시계열 DB에서 old 데이터를 삭제
- 시계열 DB에 저장되는 기간은 6개월이 바람직하나, 현장의 Data 발생량과 디스크 용량에 따라 달라짐
- 시계열 DB의 DB 백업은 별도 자료를 참조

관련 명령어

```
admin@gateway:~$ docker exec -it tsDB bash                    # 시계열 DB 접속
machbase@tsDB:~$ machsql                                      # machsql 실행

Mach> select count(*) from tag                                # 저장된 데이터 수 확인
Mach> delete from tag before TO_DATE('2020-10-15 06:41:43 000:000:000')
                                                                # tag 데이터 삭제 명령
Mach> delete from tag rollup before TO_DATE('2020-10-15 06:41:43 000:000:000');
                                                              # tag rollup 데이터 삭제 명령
```

주의 사항 : 저장되는 OPCUA 데이터의 시간은 UTC 기준이므로 삭제 시 TO_DATE 시간도 UTC 기준이어야 함 (KST보다 9시간 느림)

04. 에지 게이트웨이 운영

◆ 디스크 용량 관리-3

USB drive 마운트

- Host에 USB drive 를 mount할 dirctory를 생성(최초 설 치 사전작업) : mkdir /home/admin/exDisk
- tsDB container 최초 구동시 – mount 옵션 적용

  --mount type=bind, source=/home/admin/exDisk, target=/home/machbase/backupVolume, bind-propagation=shared
- (--mount 없이 container구동한 경우 sharedVolume의 sub directory 생성해서 mount후 container 재시작)
- USB 드라이브 연결 후 아래 명령어로 mount
- USB 드라이브 연결 후 umount

관련 명령어

```
admin@gateway:~$ sudo fdisk -l
admin@gateway:~$ sudo mount /dev/sdc1 /home/admin/exDisk
admin@gateway:~$ sudo umount /dev/sdc1
```

**

- Backup shell script(ex. fullDbBackup.sh, incDbBackup.sh)내의 Path에 /home/machbase/backupVolume 이 적용되어야 함

KOSMO KOREA SMART MANUFACTURING OFFICE

05

첨부자료

KOREA SMART MANUFACTURING OFFICE

05. 첨부자료

◆ 첨부: 시계열 DB 메모리 사용 제한

시계열 DB 메모리 사용 제한

- 마크베이스사 시계열DB 가 사용하는 최대 메모리 사이즈를 제한 할 수 있음
- 컨테이너 tsDB에 접속해 실행
- DB 테이블스페이스을 (재)생성해야 하므로 tag 테이블 생성 전에 조정해야 함
- 아래 예는 시스템 설치메모리가 8GB일 경우 최대사이즈를 7G로 제한하는 예임

1) conf 디렉토리로 이동

```
cd machbase/conf
```

2) machbase.conf 파일의 설정값 조정

```
PROCESS_MAX_SIZE = 7516192768 # 7GB        # Default 16GB → 7GB 변경
TAG_DATA_PART_SIZE = 8388608 # 8MB        # Default 16MB → 8MB
TAG_PARTITION_DEFAULT_COUNT = 2           # Default  4 → 2
```

3) machbase 및 MWA 종료

```
machadmin -s
MWAserver stop
```

4) Tablespace 재생성

```
machadmin -d
machadmin -c
```

5) machbase 및 MWA 구동

```
machadmin -u
MWAserver start
```

6) TAG 테이블 재생성하여 테스트 수행

05. 첨부자료

◆ 첨부: iptables 설정

Linux host의 iptables설정

- 리눅스 배포판 default 설정에 따라 필요한 port가 막혀 있으면 다음과 같이 port를 open해주어야 함

관련 명령어

```
admin@gateway:~/projects$ sudo iptables -I INPUT -p tcp --dport <port번호> -j ACCEPT
                                                          # 접속포트 허용 설정
admin@gateway:~/projects$ sudo netfilter-persistent save  # 접속포트 허용 적용
```

예 : iptables 설정 예

```
admin@gateway:~/projects$ sudo iptables -I INPUT -p tcp --dport 5000 -j ACCEPT
admin@gateway:~/projects$ sudo iptables -I INPUT -p tcp --dport 22 -j ACCEPT
admin@gateway:~/projects$ sudo netfilter-persistent save
```

05. 첨부자료

◆ 첨부: 시스템 TCP 성능 튜닝

Linux host의 iptables설정

- 컨테이너 내부의 TCP 세션이 많은 경우 (ex. tsDB Restful API 세션) tcp 컨널 옵션을 변경하여 성능을 개선
- 컨테이너에서 실행하지 않고 리눅스 호스트에서 실행해야 함

관련 명령어

```
admin@gateway:~/projects$ sudo sysctl -a | grep somaxcon  # TCP max connection 확인
admin@gateway:~/projects$ sudo sysctl -w net.core.somaxconn=16384 # TCP max connection 설정
admin@gateway:~/projects$ sudo sysctl -a |grep net.ipv4.tcp_tw_reuse # tcp_reuse 설정확인
admin@gateway:~/projects$ sudo sysctl -w net.ipv4.tcp_tw_reuse=1 # tcp_reuse  enable
admin@gateway:~/projects$ sudo sysctl -a |grep net.ipv4.tcp_tw_recycle # tcp_recycle 설정확인
admin@gateway:~/projects$ sudo sysctl -w net.ipv4.tcp_tw_recycle=1  # tcp_recycle  enable
```

05. 첨부자료

◆ 첨부: 시계열 DB 프로그램(마크베이스) 업그레이드

마크베이스 업데이트

- 기존에 쌓인 DB 데이터를 계속해서 사용하도록 마크베이스의 새로운 코드를 업데이트하는 절차
- 마크베이스 데모을 종료

```
machadmin -s
```

- 마크베이스 홈 디렉토리를 다른 이름으로 변경하여 기존 데이터 보호

```
mv machbase machbase-old
```

- 새로운 machbase 홈 directory 생성

```
mkdir machbase
```

- 새 버전을 machbase 홈에 압축해제

```
tar xvfz machbase-issue3299.tgz -C machbase
```

- 새버전에 MWA package가 없을 경우 machbase-old에서 machbase로 복사

```
cp -rfp machbase-old/webadmin/ machbase
```

- 기존의 DB 데이터를 새 machbase로 복사

```
mv machbase-old/dbs/ machbase
```

- 기;존의 마크베이스 configuration을 복사

```
cp machbase-old/conf/machbase.conf machbase/conf/
```

- 마크베이스 데몬을 기동

```
machadmin -u
```

05. 첨부자료

◆ 첨부: 게이트웨이 컨테이너 run 명령 예

코어 어플리케이션 컨테이너 구동 예

- 게이트웨이 시스템에서 구동되어야 하는 5개 컨테이너를 아래 옵션명령으로 run
- 아래 명령어 옵션의 컨테이너 이미지는 적절하게 변경되어야 함

```
docker run -it --name broker --hostname broker --network internalNet --restart=always eclipse-mosquitto:1.6.12

docker run -it -p 5001:5001 --name tsDB --hostname tsDB --network internalNet --restart=always -v ~/sharedFolder
:/home/machbase/sharedVolume --mount type=bind,source=/home/admin/exDisk,target=/home/machbase/backup
Volume,bind-propagation=shared machbase/machbase:6.1.15

docker run -d --name monitor -e GLANCES_OPT="-w" -v /var/run/docker.sock:/var/run/docker.sock:ro --pid host --n
etwork host --restart=always  -it nicolargo/glances:3.1.6.1

docker run -it --name opcuaModule --hostname opcuaModule -p 4840:4840 --network internalNet -v ~/sharedFold
er:/project/sharedVolume --restart=always --entrypoint '/bin/bash' nestfield/opcuamodule:210127

docker run -it --name controlModule --hostname controlModule --network internalNet -v ~/sharedFolder:/project/s
haredVolume --restart=always --entrypoint '/bin/bash' nestfield/controlmodule:rel0126
```

KOSMO KOREA SMART MANUFACTURING OFFICE

스마트공장은 스마트 경제로 가는 첫걸음입니다.

KOREA SMART MANUFACTURING OFFICE

AAS 교육자료 관리 사이트: https://github.com/kosmo-nestfield