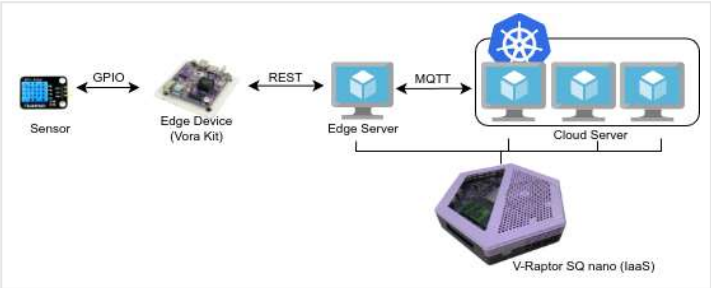
#28 [가이드] Vora Kit & V-Raptor SQ nano기반 K8S를 활용한 Edge Computing 1일 전

황원재 @hwj 변경 이력

Vora Kit & V-Raptor SQ nano기반 K8S를 활용한 Edge Computing

1. 개요

"Vora Kit & V-Raptor SQ nano기반 K8S를 활용한 Edge Computing"의 **센서**에서부터 **클라우드**까지 **데이터 전송하는 과정**은 아래 그림과 같습니다.

- 

1.1 인프라 아키텍쳐

이 가이드에서는 Vora Kit 및 V-Raptor SQ nano를 사용합니다.

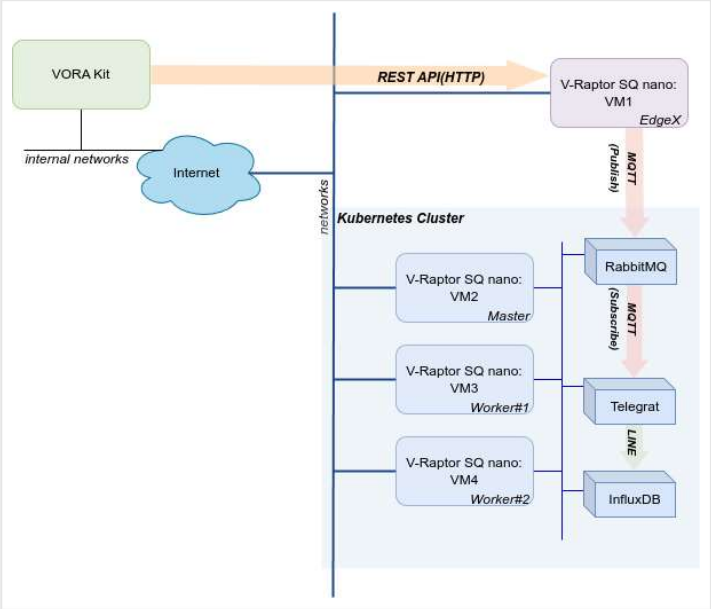
- Vora Kit은 온/습도 센서를 이용해 데이터를 측정하고,
- V-Raptor SQ nano은 사용한 총 4개의 가상머신을 이용하여, 엣지서버 및 클라우드서버(Kubernetes 인프라환경)를 구축합니다.

인프라 환경은 다음과 같습니다.

새 글쓰기

VM	서버	설명
Vora Kit	센서 서버	센서가 장착된 서버활용
SQnano VM1	엣지 서버	Edge 컴퓨팅 서비스 서버활용
SQnano VM2	클라우드 서버	K8S 마스터 서버활용
SQnano VM3	클라우드 서버	K8S 워커 서버활용
SQnano VM4	클라우드 서버	K8S 워커 서버활용

1.2 데이터 흐름도

- 

2. V-Raptor SQ nano의 가상머신

> V-Raptor SQ nano 사용법 참고 :
> [https://yona.xslab.co.kr/%EC%97%91%EC%84%B8%EC%8A%A4%EB%9E%A9/HOWTO-V-Raptor-SQ-nano/post/21]

엣지서버(Edge Server)는 센서서버로 부터 실시간 데이터를 받아 저장하고, 클라우드 서버는 센서테이터를 전달받는 역할을 합니다.

- 이 실습에서는 V-Raptor SQ nano의 가상머신을 이용하여 엣지서버와 클라우드서버를 구성합니다.
- 엣지서버는 EdgeX서비스를 설치할 예정이며, 해당 서비스는 Docker를 활용하여 설치할 계획입니다.

- EdgeX서비스는 센서로 부터 실시간 데이터를 받아 저장하고, 클라우드 서버에 REST API를 이용하여 센서 데이터를 전달합니다.

엣지서버와 클라우드 서버를 구성하기 위해서 총 4개의 VM을 생성합니다.

- VM의 사양은 다음과 같습니다.

vCPU	**4 Core**
Memory	4096 GB
Storage	25 GB

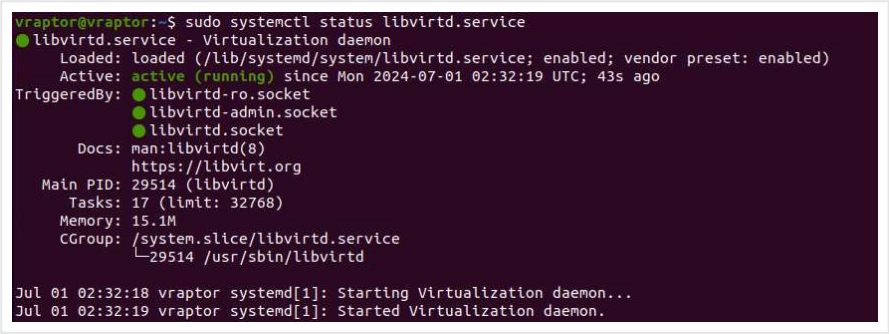
2.1. V-Raptor SQ nano 초기화

- KVM 패키지 설치

```
# apt 초기화
sudo apt update
sudo apt upgrade -y

# apt 업데이트 오류날 경우
sudo dpkg --configure -a
sudo rm -rf /var/lib/apt/lists/*
sudo apt-get update -o Acquire::CompressionTypes::Order::=g
z
sudo apt update
sudo apt-get update --fix-missing
sudo apt update; sudo apt upgrade -y

# apt 패키지 설치
sudo apt install -y qemu-kvm qemu-efi libvirt-daemon libvir
t-clients bridge-utils virtinst virt-manager
```

- 설치 확인

```
# libvirtd 설치 확인
sudo systemctl is-active libvirtd

# 사용자 그룹 추가
sudo usermod -aG libvirt $USER
sudo usermod -aG kvm $USER

# libvirtd 서비스 확인
sudo systemctl enable libvirtd --now
sudo systemctl status libvirtd.service
```



- OS 이미지 다운로드

```
sudo mkdir /img
cd /img
sudo wget https://cdimage.ubuntu.com/ubuntu/releases/20.04.5/release/ubuntu-20.04.5-live-server-arm64.iso
```

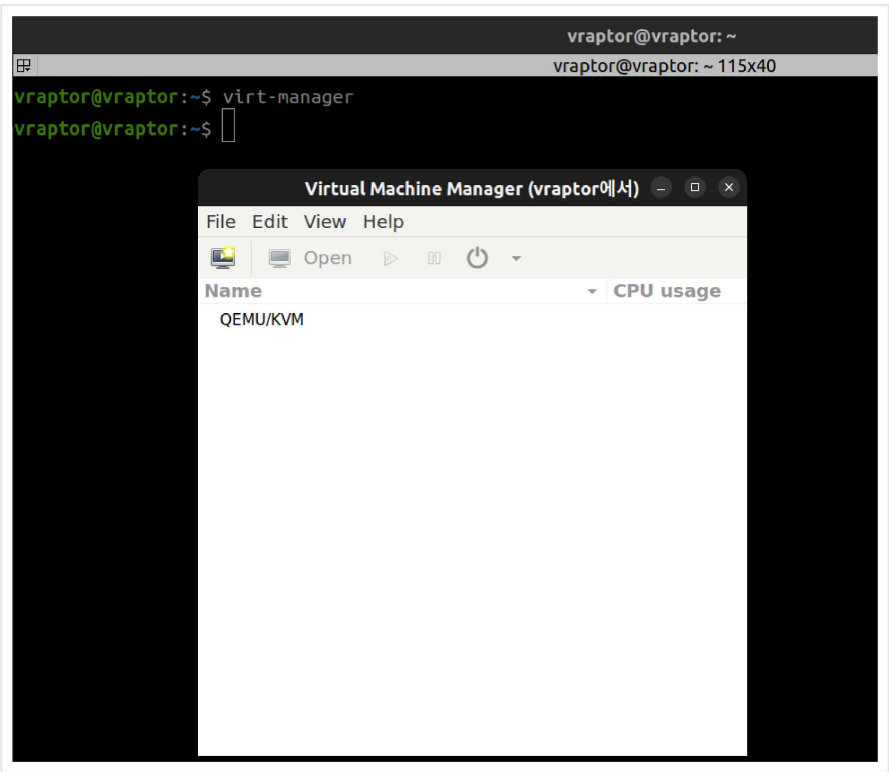
2.2. V-Raptor SQ nano에서 VM 생성

- V-Raptor SQ nano 접속 (IP가 192.168.3.140일 경우)

```
ssh vraptor@192.168.3.140 -X
```

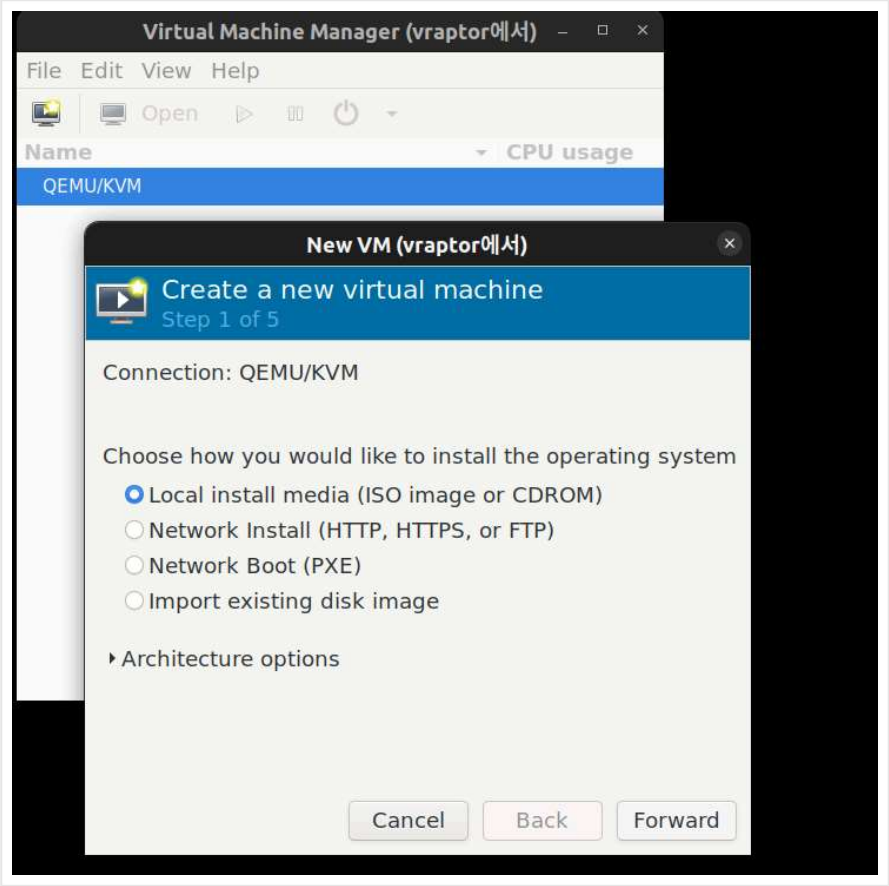
- virt-manager 실행

```
virt-manager
```



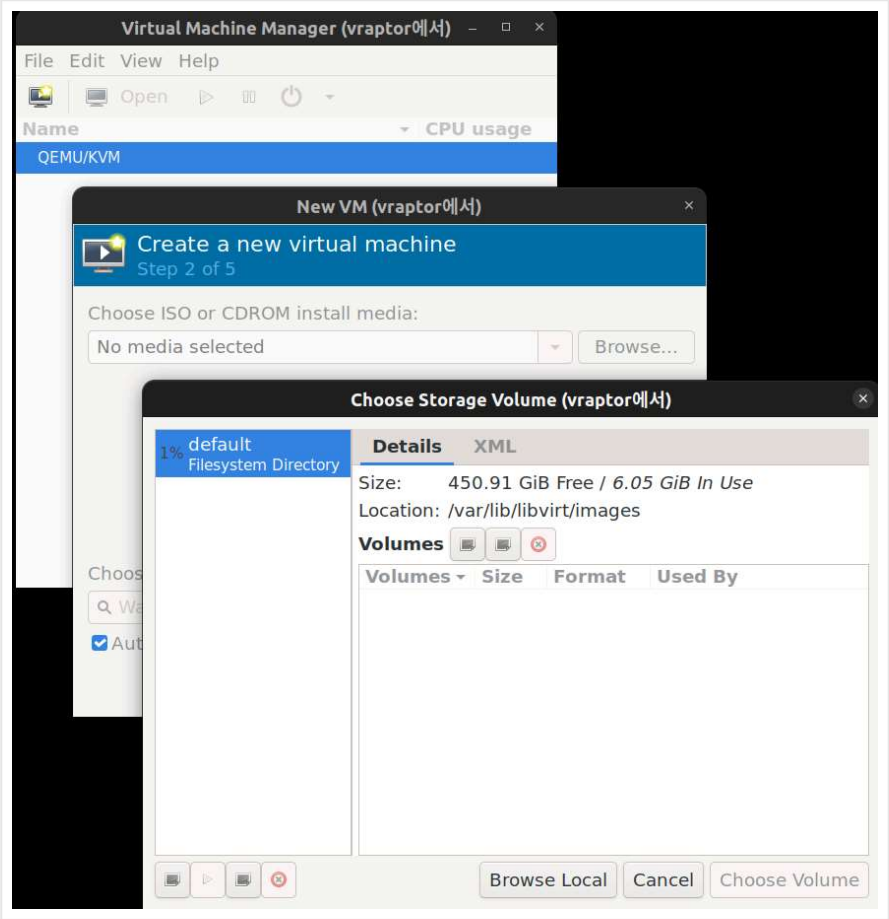
- Create a new virtual machine

새 글쓰기

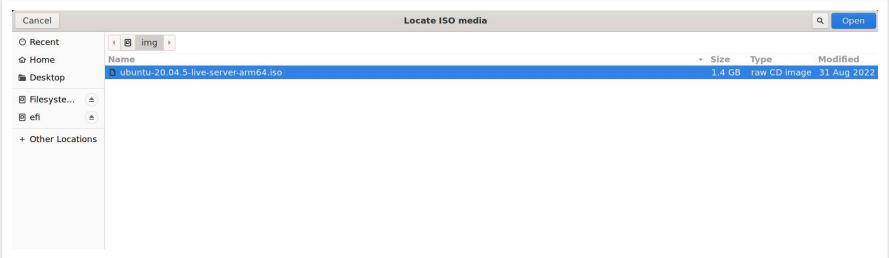


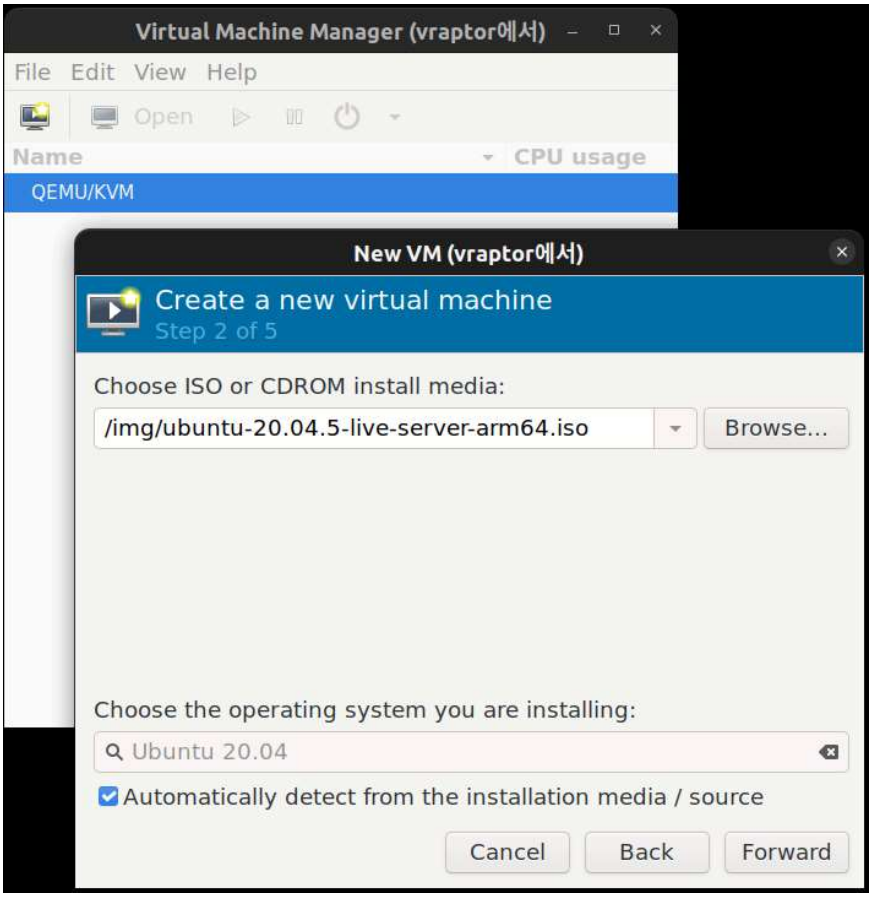
새 글쓰기

- Foward 클릭 > Browse... 클릭 > Browse Local 클릭



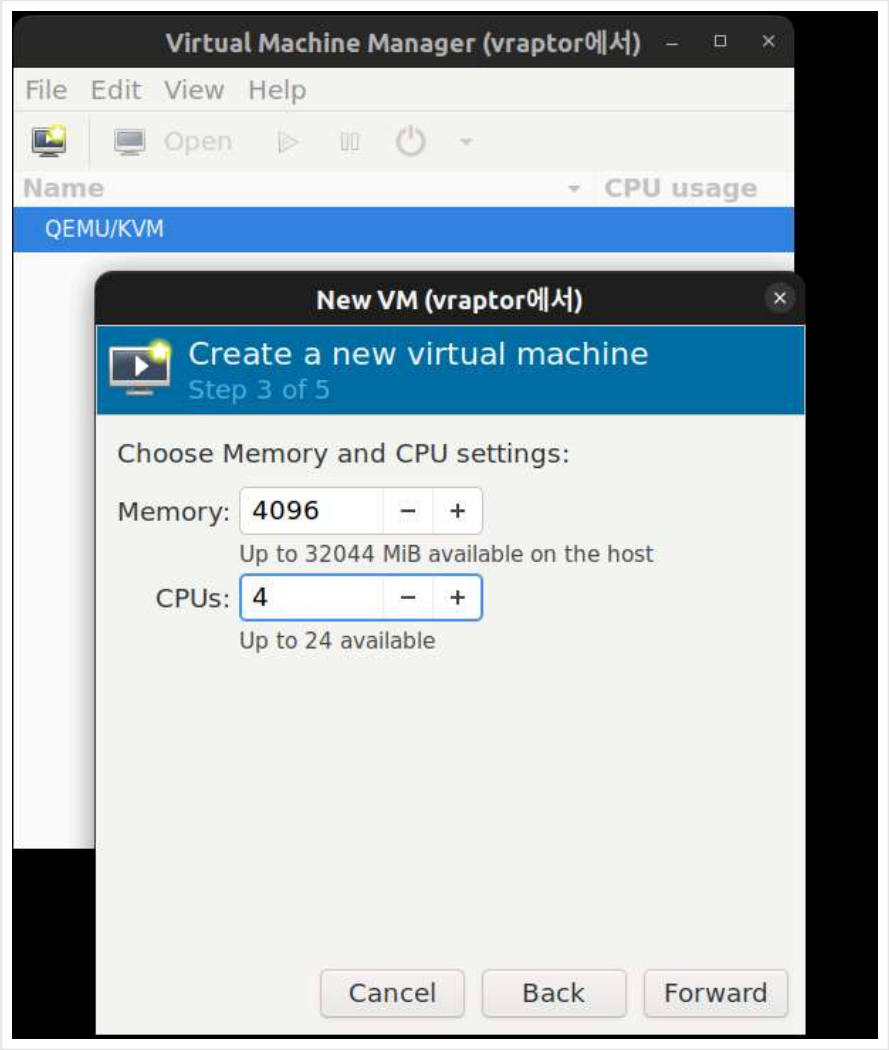
- /img 디렉토리의 ISO 파일을 Open > Forwad 클릭



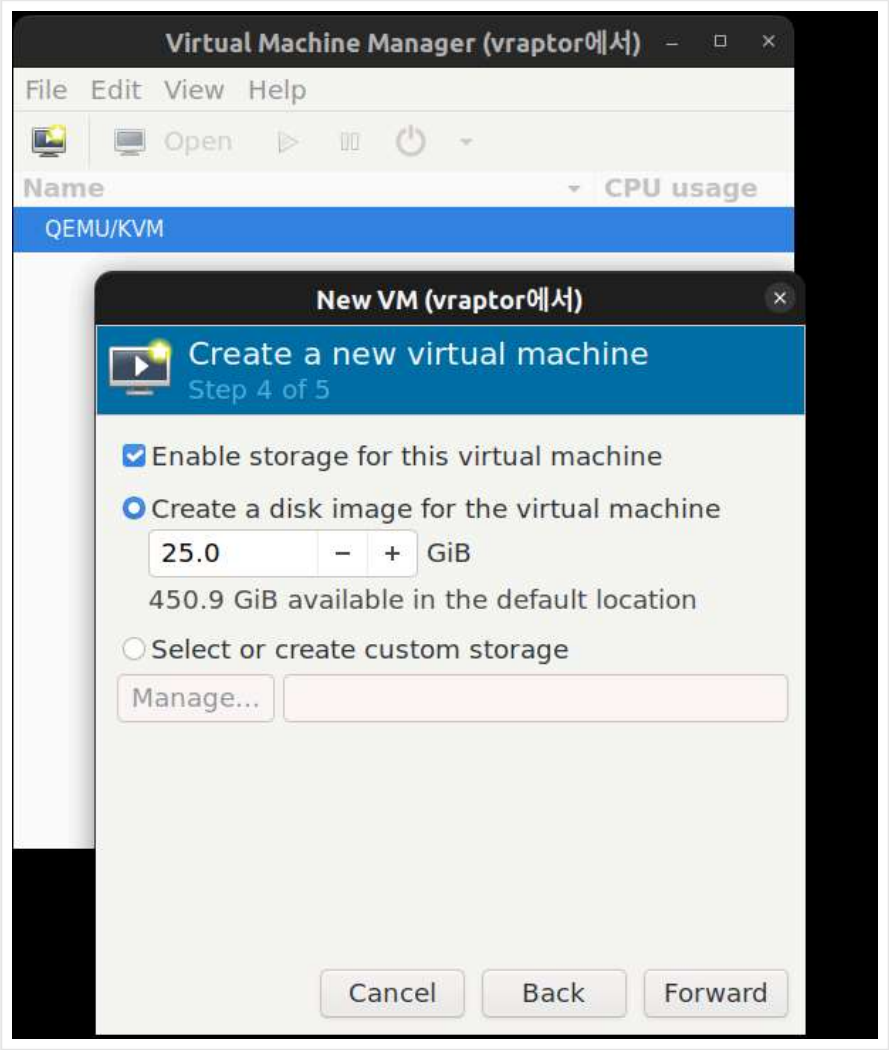

Virtual Machine Manager (vraptor에서)
File Edit View Help
Open
Name
CPU usage
QEMU/KVM
New VM (vraptor에서)
Create a new virtual machine
Step 2 of 5
Choose ISO or CDROM install media:
/img/ubuntu-20.04.5-live-server-arm64.iso
Browse...
Choose the operating system you are installing:
Ubuntu 20.04
Automatically detect from the installation media / source
Cancel
Back
Forward

새 글쓰기

- vCPU 4 core, Memory 4096 Mib 선택 > Forward 클릭



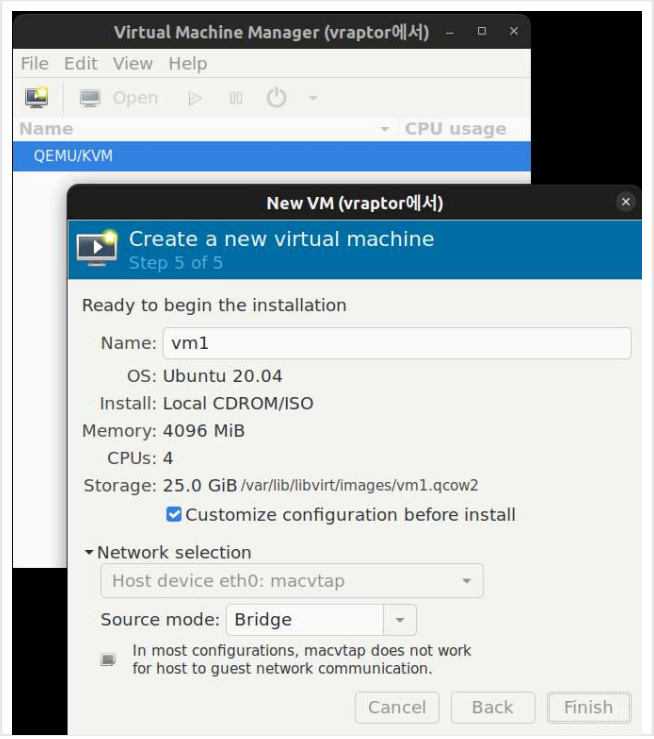
- Storage 25 GiB 선택 > Forward 클릭



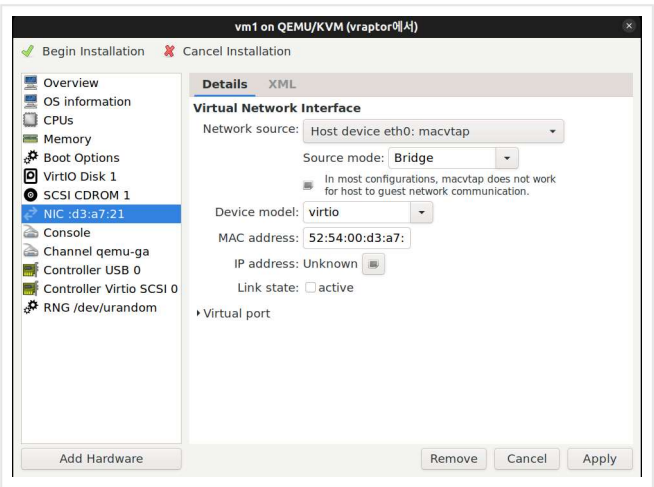
- Name을 작성 (예: VM1)
  - "Customize configuration before install" 체크
  - "Network Selection은 Host device eth0: ..." 선택
  - "Source mode"는 Bridge 선택

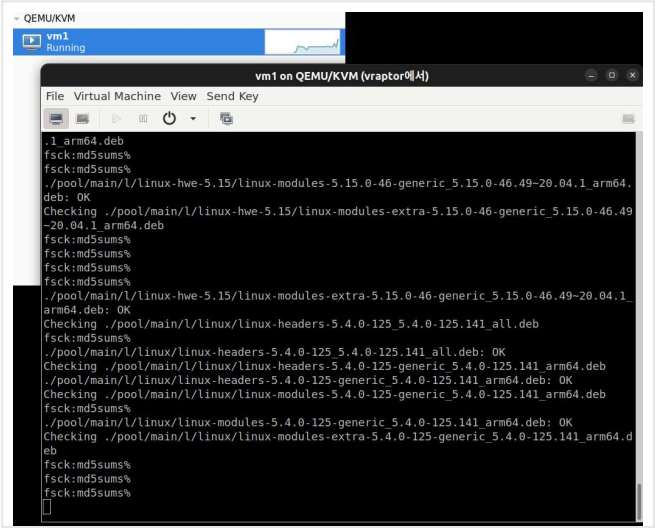
새 글쓰기

- Finish 클릭



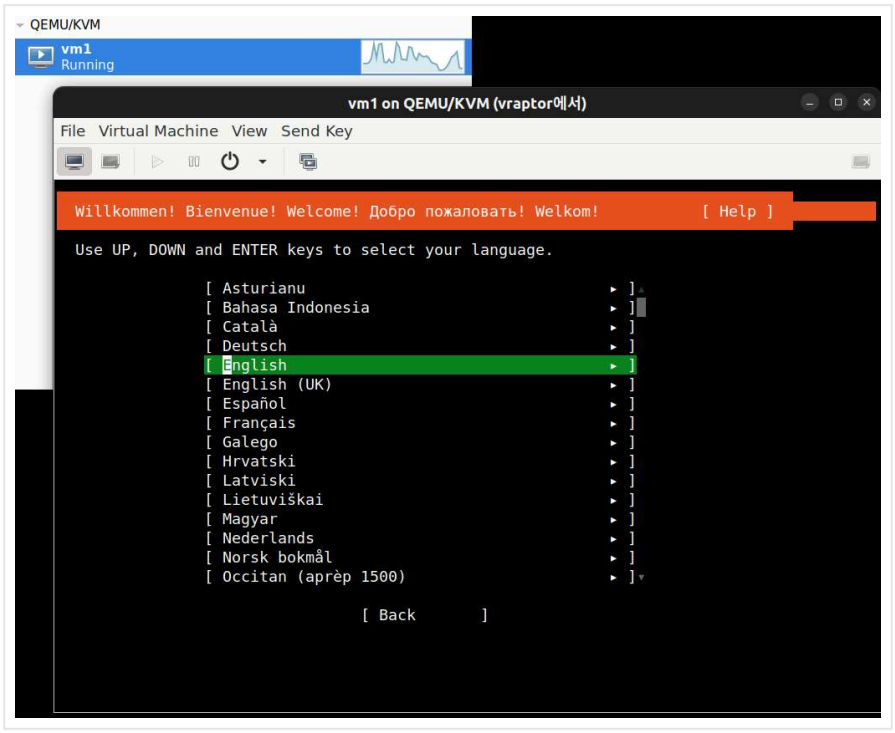
- **(중요!!)**: OS 설치시 네트워크 인터페이스 끄기
  - NIC : ... 탭 >
  - "acive" 체크 **해제** >
  - Apply 클릭 >
  - Begin Installation 클릭



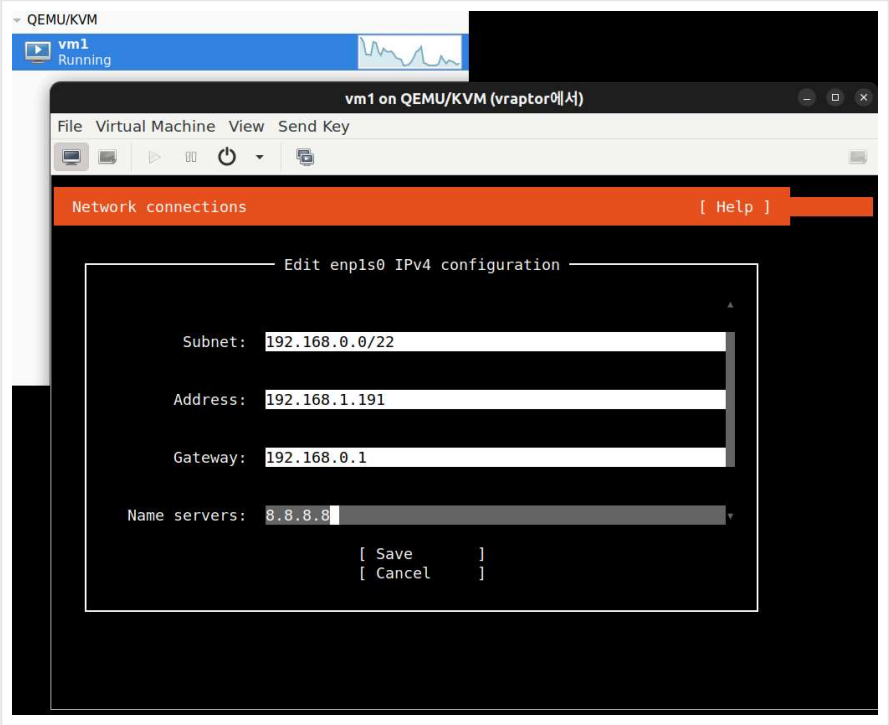


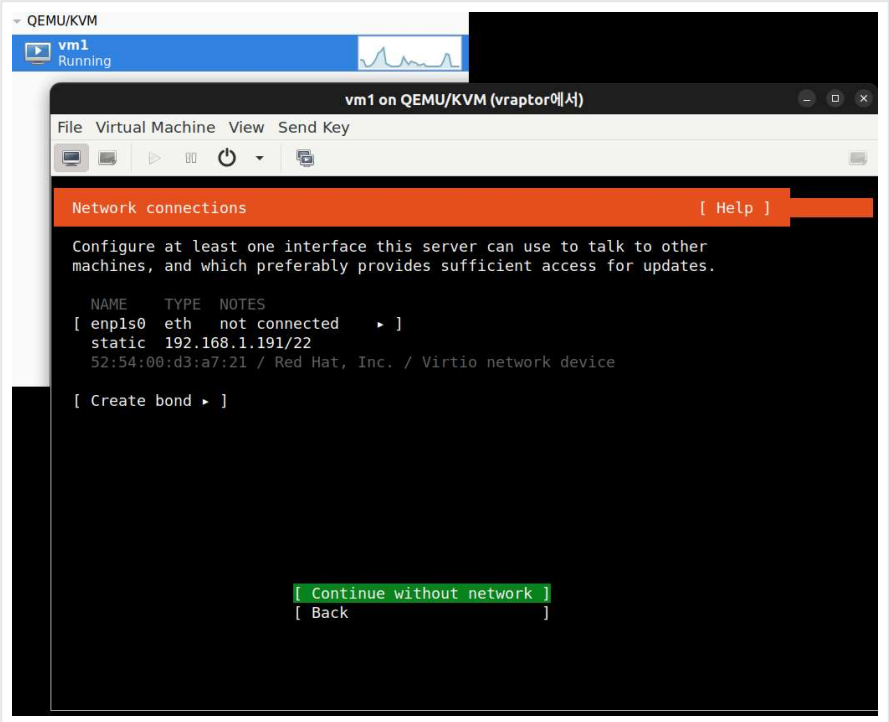
새 글쓰기

- Ubuntu OS 설치 프로세스 진행



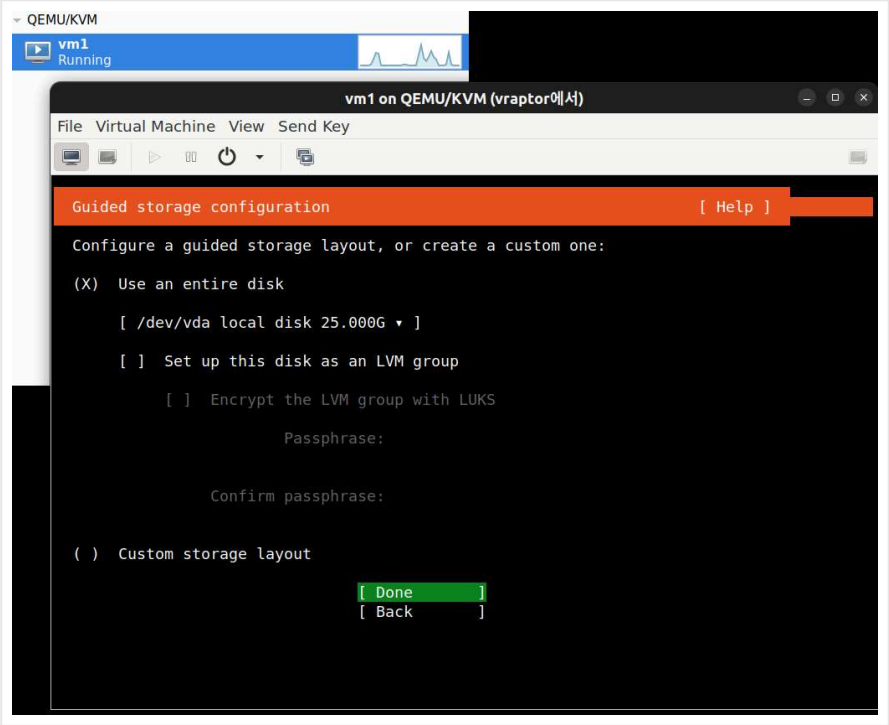
- 인터넷 안되는 상태에서 네트워크 설정 (Continue without network)



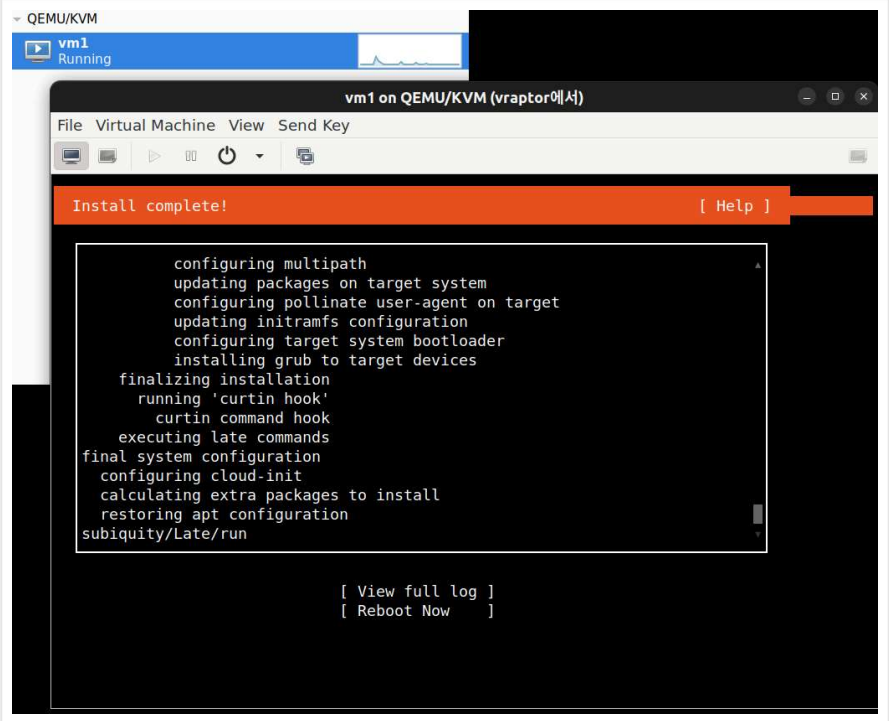


새 글쓰기

- Set up this disk as an LVM group 체크 해제 (OpenSSH 설치도 체크 안함)

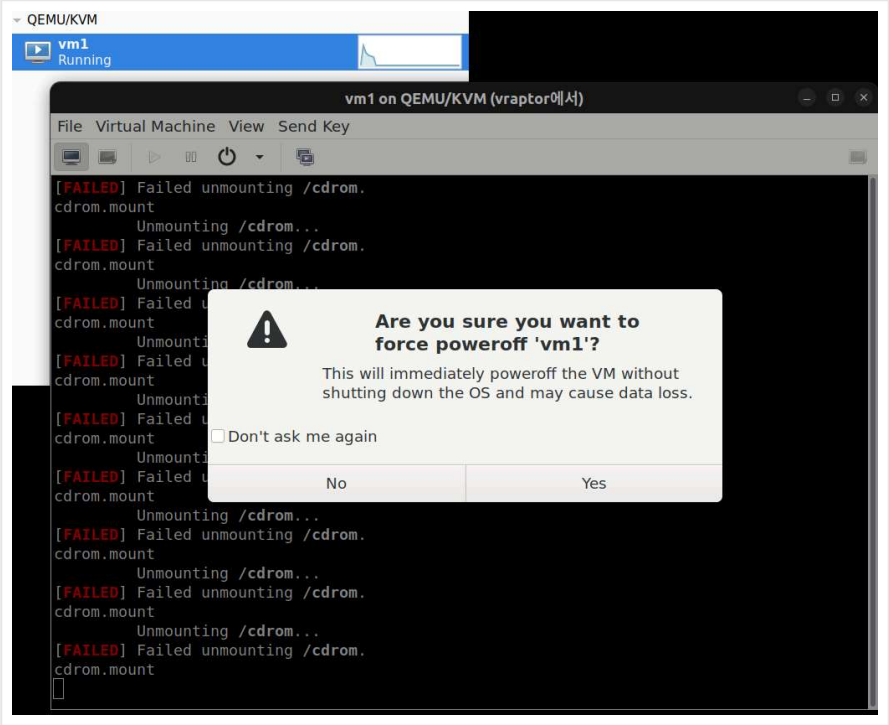


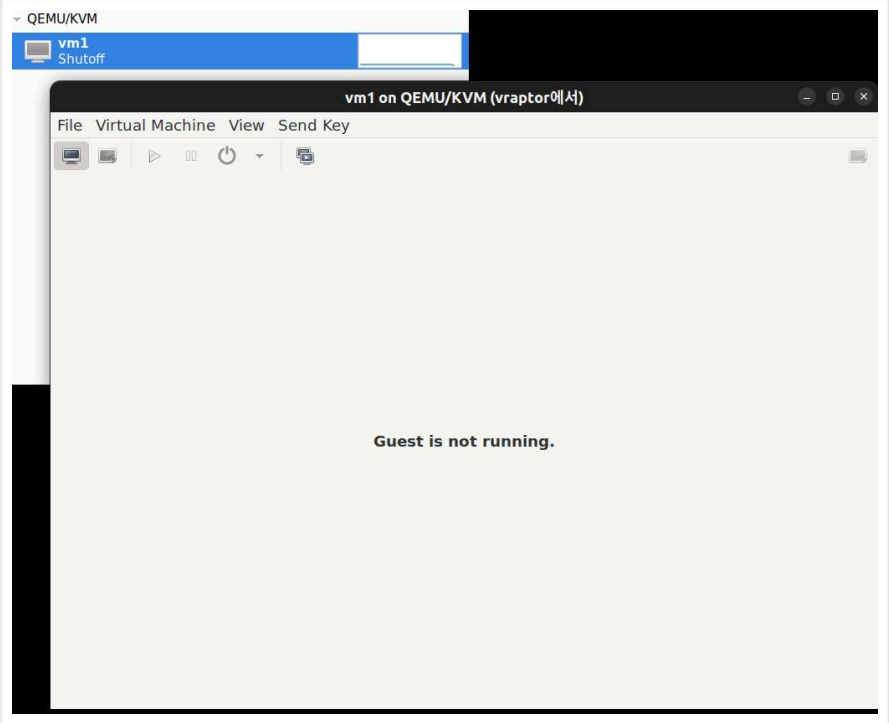
- 설치가 완료되면 Reboot Now 클릭



새 글쓰기

- 전원 버튼 옆 아래 화살표 눌러서 “Force off” 클릭

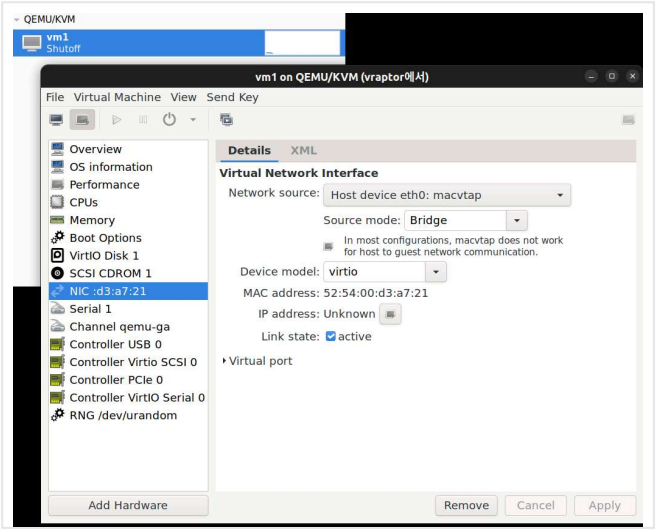




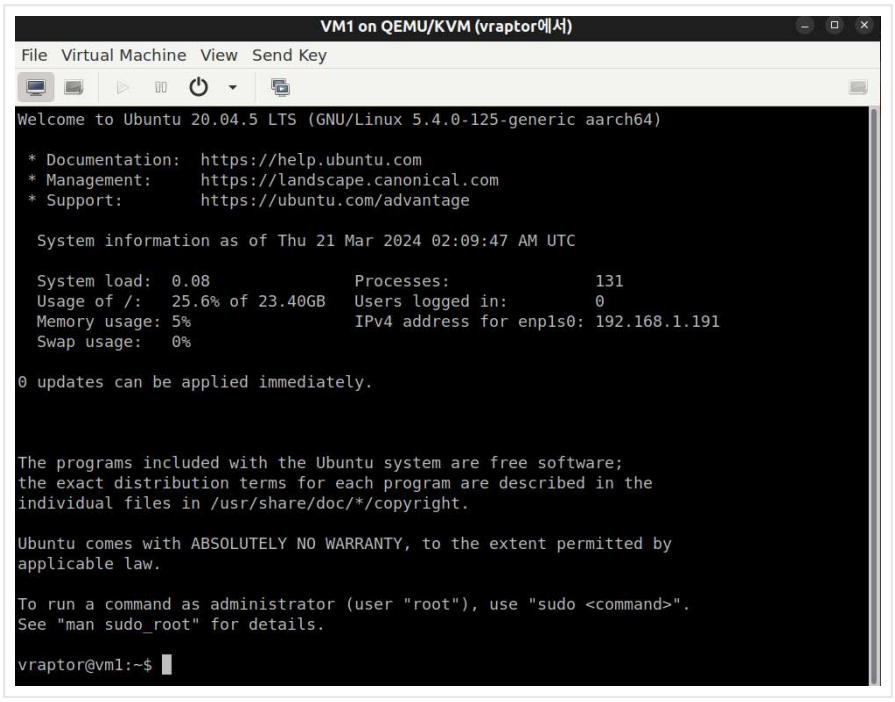
- show virtual hardware details 클릭
  - NIC탭 > "active" 체크
  - Apply 클릭

새 글쓰기

- Power on 클릭



- 로그인 및 네트워크 정상동작 확인



2.3. V-Raptor SQ nano에서 VM의 환경설정

- 총 4개의 VM을 만들고, 그 정보는 아래와 같습니다.

Node	Info.	IP	vCPU	Memory(GB)	Storage(GB)
VM1	Edge Server	192.168.1.191	4	4096	25
vm2	K8S Master	192.168.1.192	4	4096	25

새 글쓰기

Node	Info.	IP	vCPU	Memory(GB)	Storage(GB)
vm3	K8S Worker#1	192.168.1.193	4	4096	25
vm4	K8S Worker#2	192.168.1.194	4	4096	25

- ssh 설치 및 설정

```
sudo apt install openssh-server -y
sudo sed -i 's/#PasswordAuthentication yes/PasswordAuthentication yes/' /etc/ssh/sshd_config
sudo systemctl restart ssh
```

3. [센서서버]에서 [엣지서버]로 센서데이터 전송 설정

3.1. 엣지서버에서 EdgeX서비스 실행

- 엣지서버 접속

```
ssh vraptor@192.168.1.191
```

- 호스트 네임 변경

```
sudo hostnamectl set-hostname vm1
```

- Docker 설치

```
git clone https://yona.xslab.co.kr:443/%EC%97%91%EC%84%B8%EC%8A%A4%EB%9E%A9/HOWTO-V-Raptor-SQ-nano
cd HOWTO-V-Raptor-SQ-nano/edgex
chmod +x docker-install.sh
sudo ./docker-install.sh
```

- 예제파일에서 docker-compose.yml 파일을 실행시켜, EdgeX 컨테이너를 실행한다.

```
sudo docker compose up -d

sudo docker compose ps -a
```

새 글쓰기

- (옵션) root권한이 아닐경우, Docker가 runnigs 오류 발생합니다. 아래 명령어 실행시 해결가능합니다.

```
ls -al /var/run/docker.sock
sudo groupadd docker
sudo usermod -aG docker $USER
sudo chown root:docker /var/run/docker.sock
sudo service docker restart
sudo chmod 666 /var/run/docker.sock
```

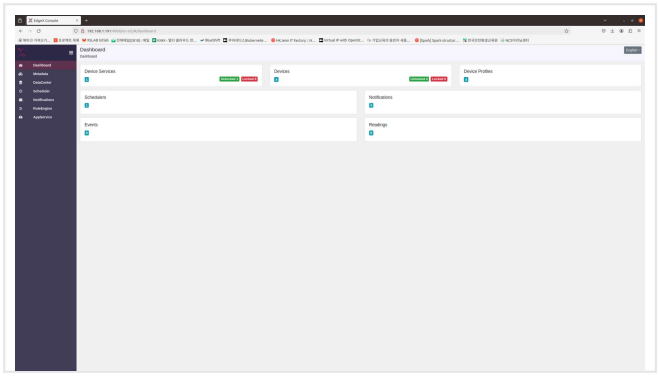
- 엣지서버의 필요한 포트
  - 4000 : Edge X 웹 인터페이스용 포트
  - 8500 : Consul 웹 인터페이스용 포트
  - 9082 : eKuiper 웹 인터페이스용 포트
  - 59880 : 데이터 서비스 RESTful API용 포트
  - 59881 : Metadata 서비스용 포트
  - 59882 : Command 서비스용 포트
  - 48101 : MQTT 앱 서비스용 포트
  - 59986 : Device REST 포트

3.2. EdgeX 웹 서비스 접속

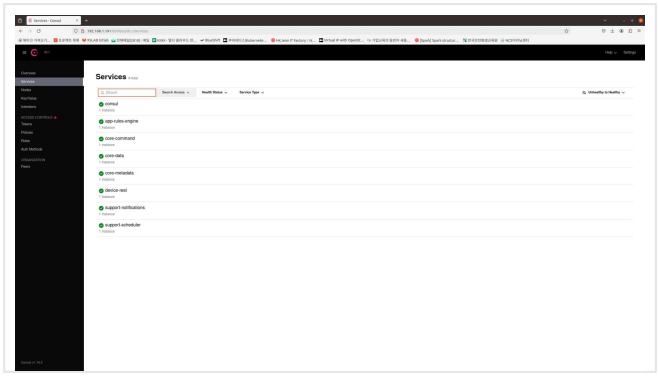
웹 브라우저로 다음의 주소(vm1의 ip)에 접속합니다.

- Edge X Dashboard
  - http://192.168.1.191:4000

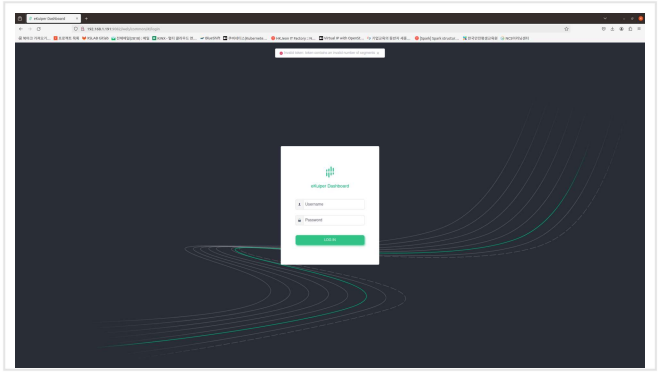
새 글쓰기



- Consol Dashboard
  - http://192.168.1.191:8500



- eKuiper Dashboard
  - http://192.168.1.191:9082
    - ID : admin
    - P/W : public



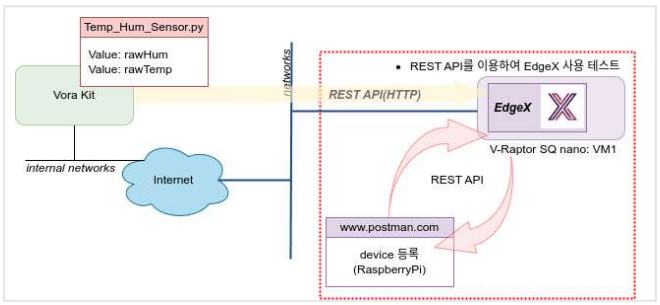
3.2. EdgeX서비스에 센서등록 및 데이터 전송확인

엣지서버에는 EdgeX서비스가 동작합니다.

- Postman API를 이용하여 EdgeX서비스에 센서 및 프로파일을 등록합니다.
- Postman은 REST API를 사용하기 위한 플랫폼입니다.

> Postman : https://www.postman.com/

새 글쓰기

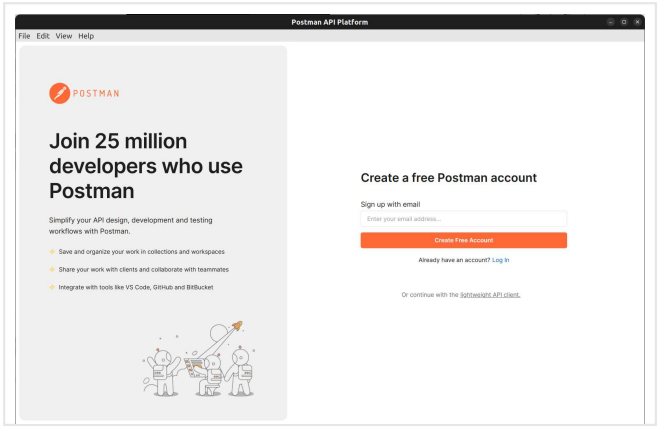
- EdgeX서비스에 센서 디바이스 정보를 등록하기 위해서는 REST API를 이용합니다. 따라서 Postman 플랫폼을 이용하여 REST API를 손쉽게 사용할 수 있습니다.
  - 

- Postman 설치 및 실행
  - Postman은 별도의 PC(노트북)에서 설치하여 사용할 수 있습니다.
  - PC(노트북)을 사용할 경우

```
sudo snap install postman
```

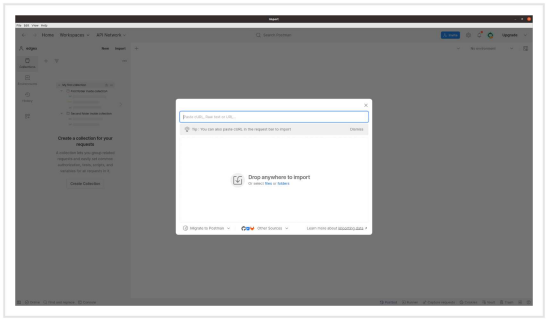
- Postman 로그인 및 워크스페이스 실행

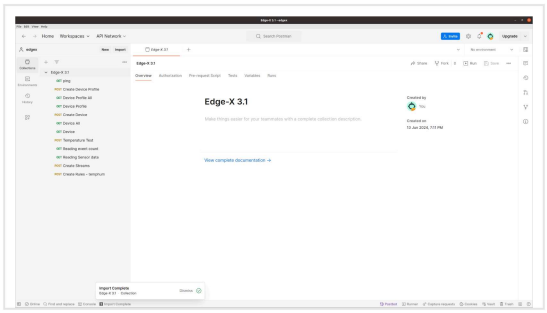
> 워크스페이스 자세한 내용은 여기를 참고.

  - 

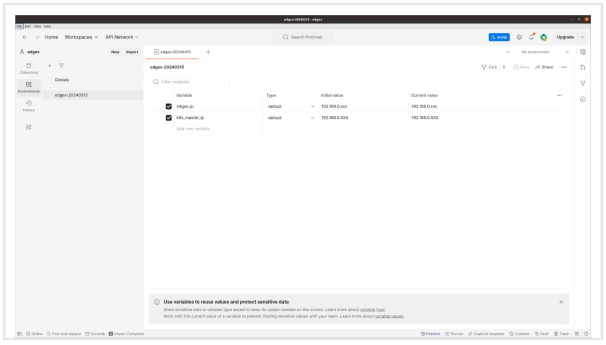
  - My Workspace 생성
  - Collections 탭 > import
    - ...collection.json 파일을 import 합니다.

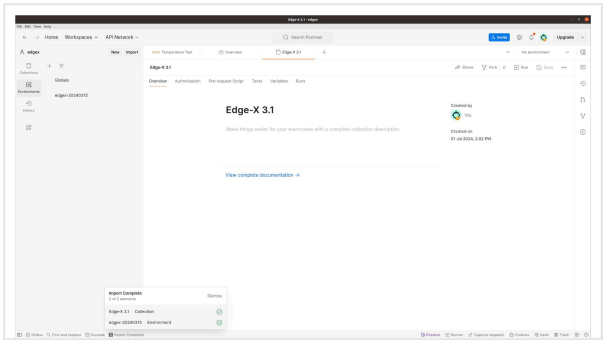
- Edge-X_3.1.postman_collection.json





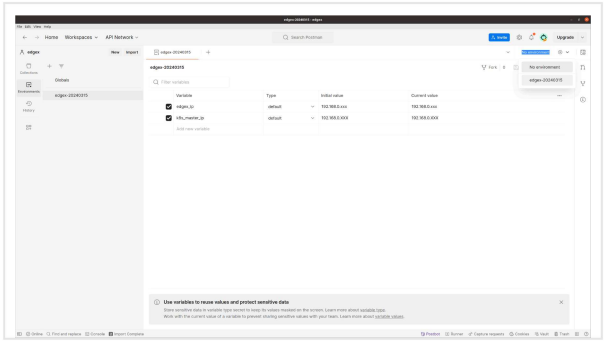
- Environments 탭 > import
  - ...Environments.json 파일을 import 합니다.
    - Edge-X_3.1.postman_environment.json



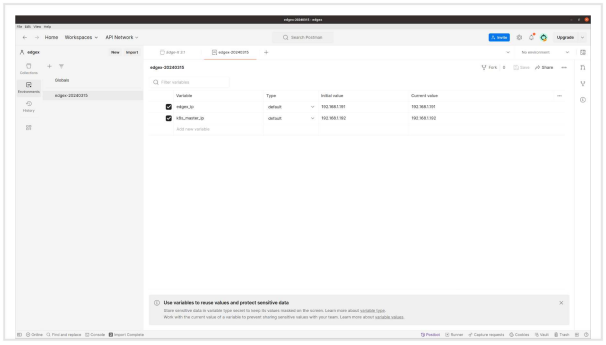


- Environments 탭 > edgex-20240315을 선택합니다.

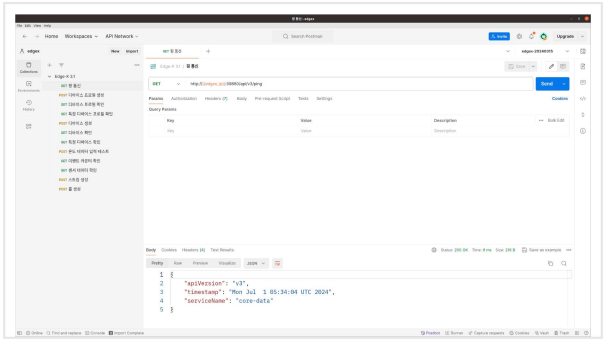
새 글쓰기



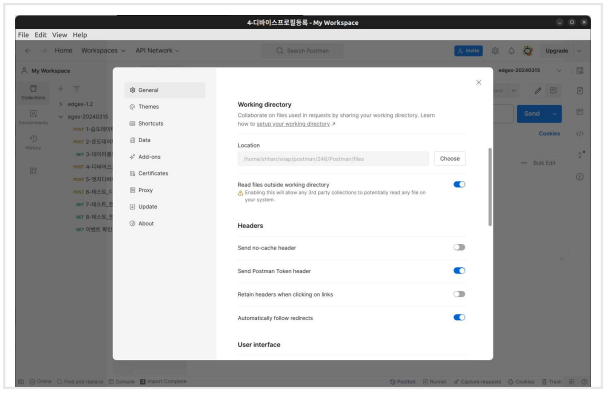
- Initial value와 Current value에 Edge X 서버와 K8s 마스터 서버의 (vm2)의 IP를 입력한 후 우측 상단의 Save를 눌러 저장합니다.



- Collections 탭 > 1번 핑 통신을 실행해 통신이 정상적으로 되는지 확인합니다.



- 톱니바퀴 설정 클릭 > Read files outside ... 체크

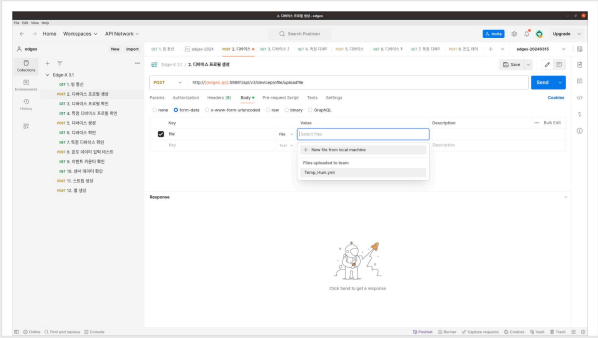


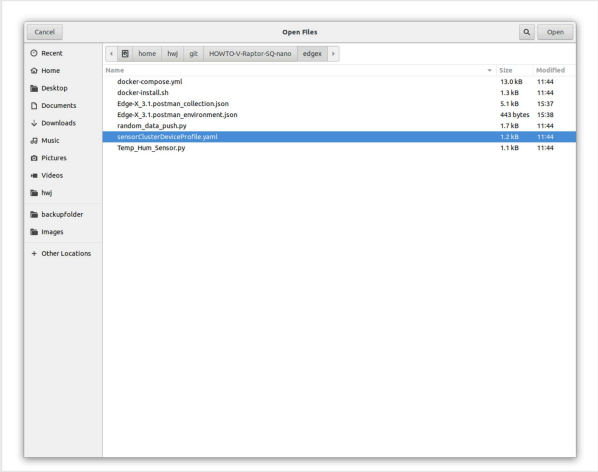
- Collections 탭 > 2번 디바이스 프로필 생성 > Body 탭의 file에서 select files를 선택한 후 sensorClusterDeviceProfile.yaml 파일을 넣

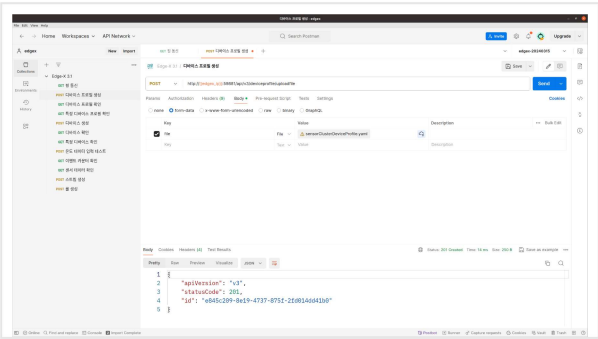
고 SEND를 실행합니다.

새 글쓰기

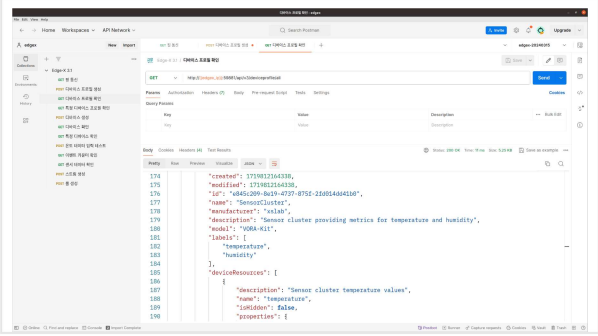
- sensorClusterDeviceProfile.yaml



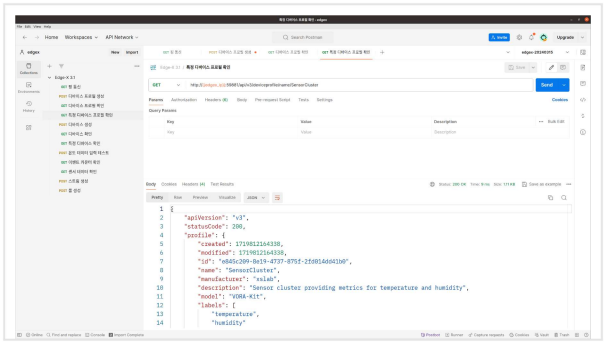




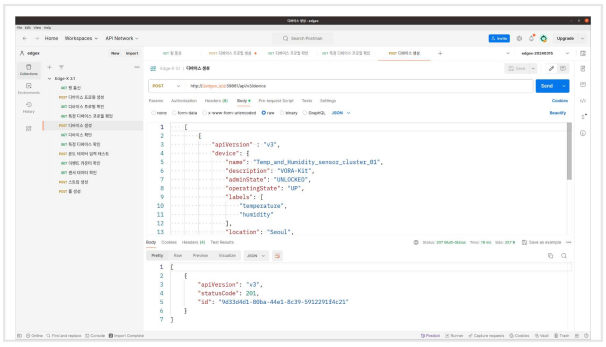
- Collections 탭 > 3, 4번을 실행해 디바이스 프로필이 정상적으로 생성되었는지 확인합니다.



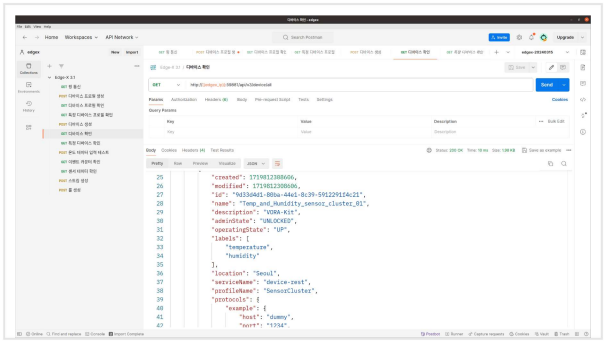
새 글쓰기

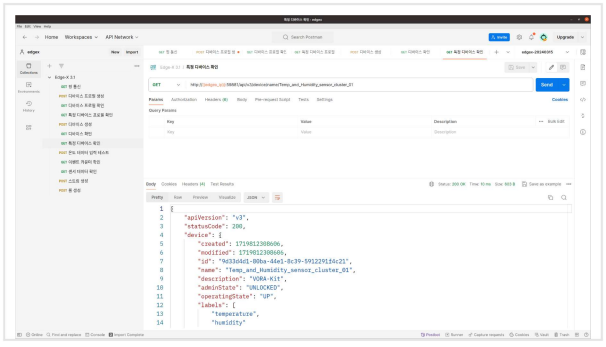


- Collections 탭 > 5번을 실행해 디바이스를 생성합니다.

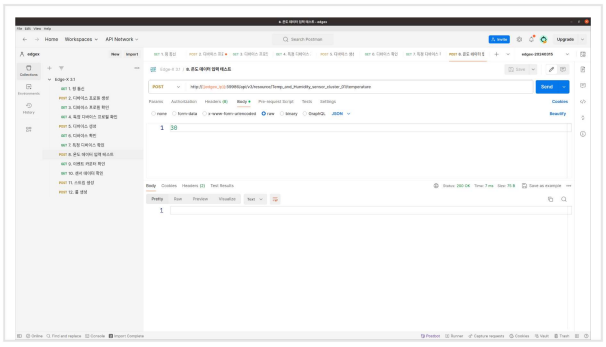


- Collections 탭 > 6, 7번을 실행해 디바이스가 정상적으로 생성되었는지 확인합니다.

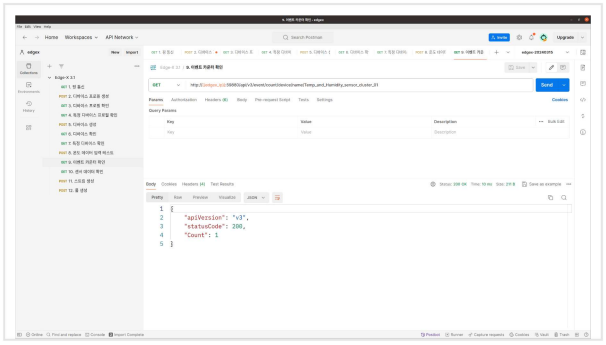




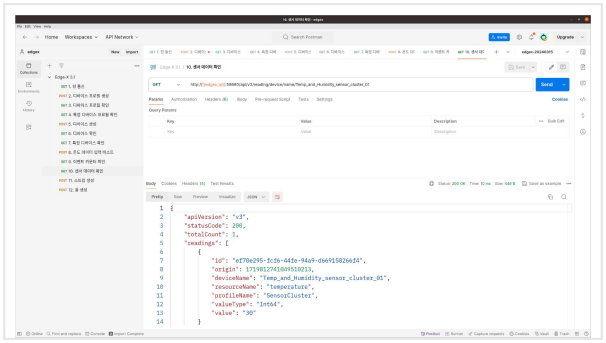
- Collections 탭 > 8번을 실행해 온도데이터를 전송합니다.



- Collections 탭 > 9번을 실행해 센서 데이터의 카운터가 증가했는지 확인합니다.



- Collections 탭 > 10번을 실행해 센서 데이터를 확인합니다.



3.3. 센서서버에서 센싱데이터 전송

랜덤으로 생성된 센서 데이터를 EdgeX에 전송합니다.

- 센서 테스트

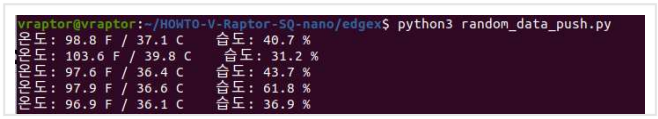
```
git clone https://yona.xslab.co.kr/%EC%97%91%EC%84%B8%EC%8A%A4%EB%9E%A9/HOWTO-V-Raptor-SQ-nano
cd HOWTO-V-Raptor-SQ-nano/edgex/
```

```
vi random_data_push.py
>
... #수정필요
# EdgeX server ip
```

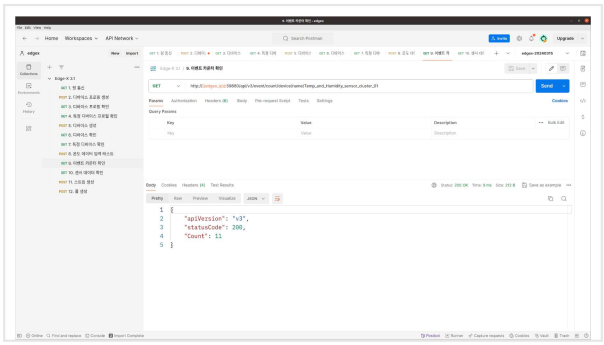
```
edgexip = "192.168.1.191"
...
```

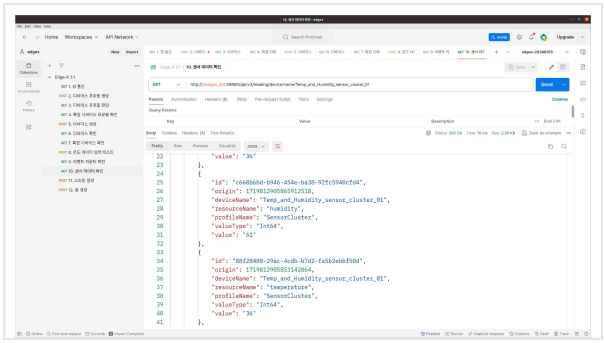
새 글쓰기

```
# 랜덤 데이터 전송
python3 random_data_push.py
```

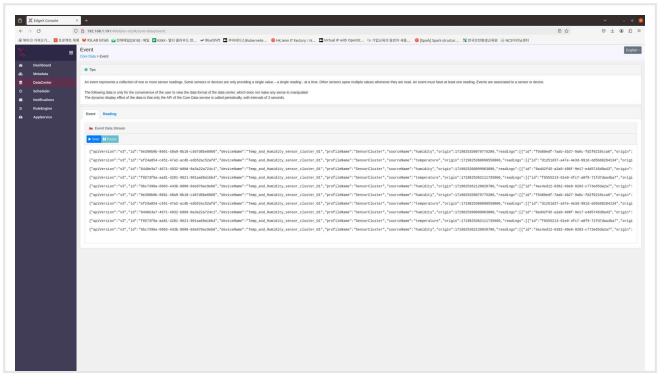
- 

- 전송한 센서데이터를 확인합니다.
  - Postman 9, 10번 활용

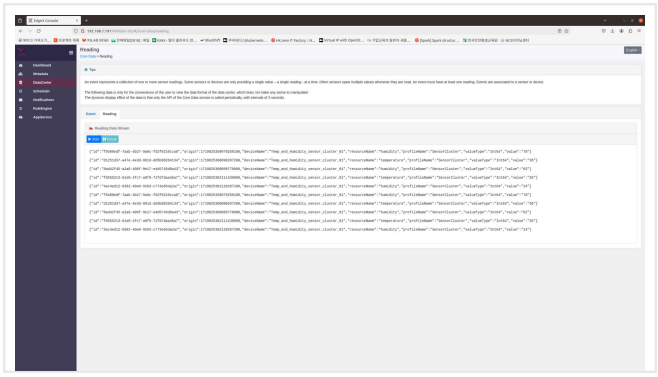




- Edge-X의 웹에서 데이터가 전송되었는지 확인가능합니다.
  - Edge-X의 웹에서 DataCenter 탭에서 전송된 데이터를 확인할 수 있습니다.



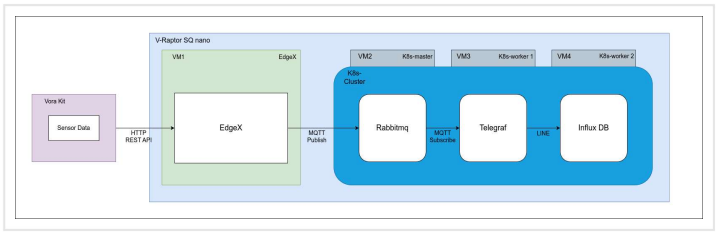
새 글쓰기



---

4. 클라우드(쿠버네티스) 서비스 구축

이번 실습에서는 V-Raptor SQ mini의 VM2~VM4를 이용하여 쿠버네티스기반 클라우드 환경을 구축합니다.
또한, 엣지서버에 저장되고 있는 센서데이터를 쿠버네티스로 구축한 클라우드서버로 데이터를 전송합니다.



5.1. 가상머신에 도커 및 쿠버네티스 설치

쿠버네티스를 설치할 VM은 다음과 같이 활용합니다.

Node	Info.	IP	vCPU	Memory(GB)	Storage(GB)
vm2	K8S Master	192.168.1.192	4	4096	25
vm3	K8S Worker#1	192.168.1.193	4	4096	25
vm4	K8S Worker#2	192.168.1.194	4	4096	25

생성된 모든 인스턴스에 아래와 같이 도커 및 쿠버네티스를 설치합니다.

- 호스트 네임 변경

```
sudo hostnamectl set-hostname <vmX>

ex) vm2의 경우
sudo hostnamectl set-hostname vm2
```

- 실습 데이터 다운로드

```
sudo apt update; sudo apt upgrade -y
git clone https://yona.xslab.co.kr/%EC%97%91%EC%84%B8%EC%8
A%A4%EB%9E%A9/HOWTO-V-Raptor-SQ-nano
cd HOWTO-V-Raptor-SQ-nano/k8s
```

- 실행파일 권한설정

```
chmod +x docker-install.sh
chmod +x k8s-install-setup.sh
```

- 스왑 영역 삭제

```
sudo swapoff -a
sudo sed -i '/\/swap.img/d' /etc/fstab
```

- 도커 및 쿠버네티스 설치

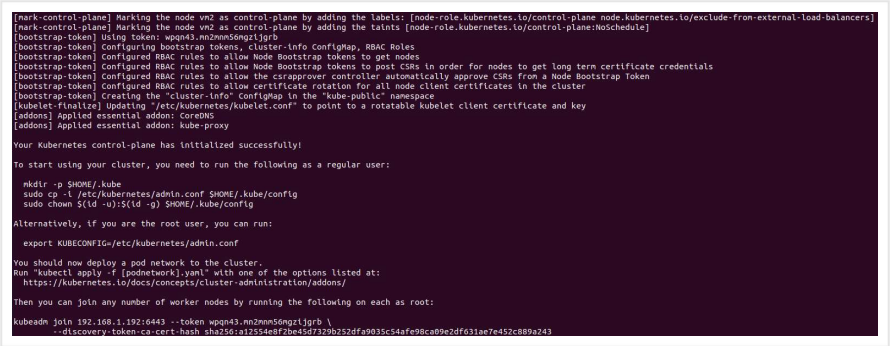
```
sudo ./docker-install.sh
sudo ./k8s-install-setup.sh
```

4.2. 쿠버네티스 클러스터 생성

- 마스터 노드인 `VM2` 에서 클러스터를 생성합니다.

```
MASTER_IP=`hostname -I | awk '{print $1}'`
echo $MASTER_IP
sudo kubeadm init --apiserver-advertise-address="${MASTER_I
P}" --pod-network-cidr "10.244.0.0/16"
```

새 글쓰기



- 마스터 노드인 `VM2`에서 사용자 권한에서 아래 명령어 실행합니다.

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

- 워커 노드인 `VM3`, `VM4`에 접속해 마스터 노드에 등록합니다.
  - 위의 명령어의 결과값을 입력합니다.

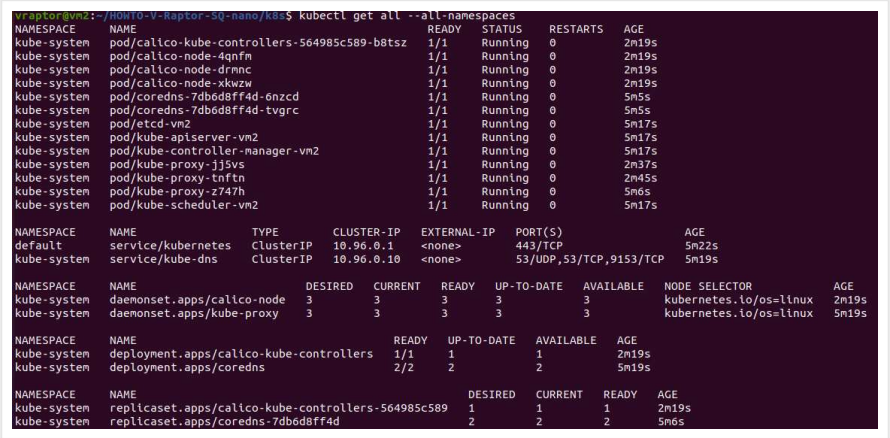
```
sudo kubeadm join 192.168.1.192:6443 --token wpqn43.mn
2mnm56mgzijgrb \
  --discovery-token-ca-cert-hash sha256:a12554e8f2be45
d7329b252dfa9035c54afe98ca09e2df631ae7e452c889a243
```

- 마스터 노드인 `VM2`에서 CNI를 구성합니다.

```
kubectl apply -f calico.yaml
```

- 마스터 노드인 `VM2`에서 K8s 시스템이 정상적으로 구성되었는지 확인합니다.

```
kubectl get all --all-namespaces
```



- 마스터 노드인 `VM2`에서 노드들을 확인합니다.

새 글쓰기

```
kubectl get nodes
```

```
vraptor@vm2:~/HOWTO-V-Raptor-SQ-nano/k8s$ kubectl get nodes
NAME   STATUS   ROLES           AGE    VERSION
vm2    Ready    control-plane   4m47s  v1.30.2
vm3    Ready    <none>          2m8s   v1.30.2
vm4    Ready    <none>          2m     v1.30.2
```

5. 클라우드(쿠버네티스) 서비스 실행

5.1. POD 서비스 실행

- 마스터 노드인 `VM2` 에서 PV(영구 볼륨)생성에 필요한 스토리지 클래스를 생성한 후 확인합니다.

```
cd ~/HOWTO-V-Raptor-SQ-nano/k8s
kubectl apply -f local-path-storage.yaml
kubectl get storageclass
```

```
vraptor@vm2:~/HOWTO-V-Raptor-SQ-nano/k8s$ kubectl get storageclass
NAME         PROVISIONER             RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
local-path   rancher.io/local-path   Delete          WaitForFirstConsumer   false                  20s
```

- 마스터 노드인 `VM2` 에서 Rabbitmq의 Deployment, Service를 생성합니다.

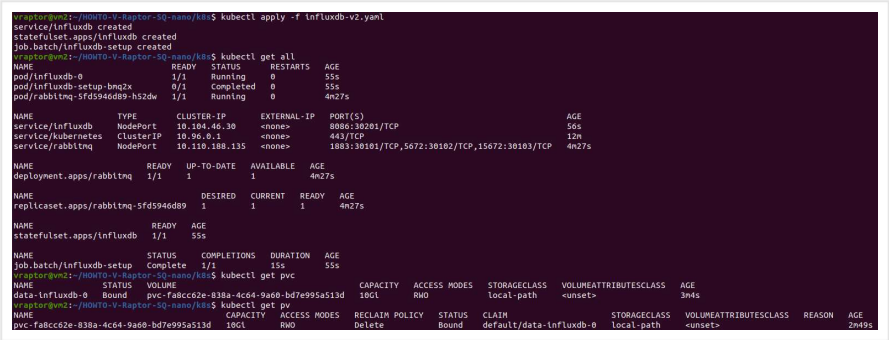
```
kubectl apply -f rabbitmq.yaml
kubectl get all
```

```
vraptor@vm2:~/HOWTO-V-Raptor-SQ-nano/k8s$ kubectl apply -f rabbitmq.yaml
service/rabbitmq created
deployment.apps/rabbitmq created
vraptor@vm2:~/HOWTO-V-Raptor-SQ-nano/k8s$ kubectl get all
NAME                            READY   STATUS    RESTARTS   AGE
pod/rabbitmq-5fd5946d89-h52dw   1/1     Running   0          56s

NAME                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                                                  AGE
service/kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP                                                  8m31s
service/rabbitmq     NodePort    10.110.188.135   <none>        1883:30101/TCP,5672:30102/TCP,15672:30103/TCP            56s

NAME                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/rabbitmq   1/1     1            1           56s

NAME                                  DESIRED   CURRENT   READY   AGE
replicaset.apps/rabbitmq-5fd5946d89   1         1         1       56s
```

- 마스터 노드인 `VM2` 에서 influxdb의 Deployment, Service,Pvc 를 생성합니다.

```
kubectl apply -f influxdb-v2.yaml
kubectl get all
kubectl get pvc
kubectl get pv
```

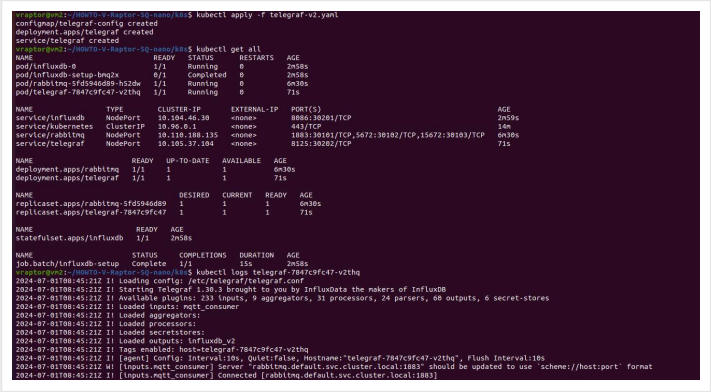
새 글쓰기



- `influxdb의 파드가 다음과 같이 Pending 상태 일 때`
  - kubectl get pvc를 실행해서 PVC의 상태를 확인합니다.
  - PVC의 상태가 Pending 상태라면 kubectl get storageclass를 실행해서 스토리지 클래스가 생성되어있는지 확인합니다.
  - kubectl delete -f influxdb.yaml, kubectl delete pvc influxdb-data-influxdb-0 로 PVC를 삭제합니다.
  - kubectl apply -f local-path-storage.yaml로 스토리지 클래스를 생성합니다.
  - kubectl apply -f influx.yaml 를 진행합니다.

- 마스터 노드인 `VM2`에서 telegraf의 Deployment, Service 를 생성합니다.

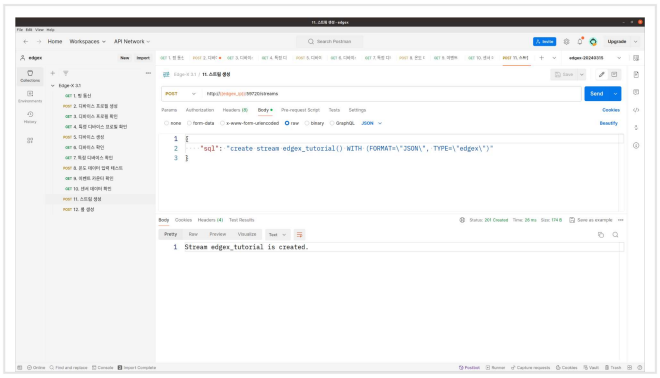
```
kubectl apply -f telegraf-v2.yaml
kubectl get all
kubectl logs telegraf-<컨테이너 이름>
```



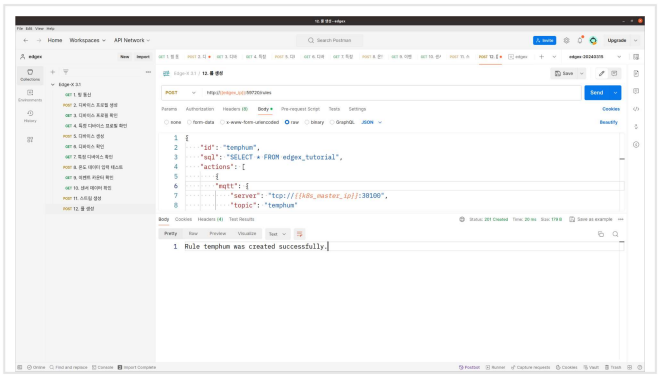
5.2. Rabbitmq MQTT 통신 세팅

- Postman에서 Collections 탭 > 11번을 실행해 MQTT 통신을 위한 스트림을 생성합니다.

새 글쓰기



- Collections 탭 > 12번을 실행해 MQTT 통신을 위한 룰을 생성합니다.

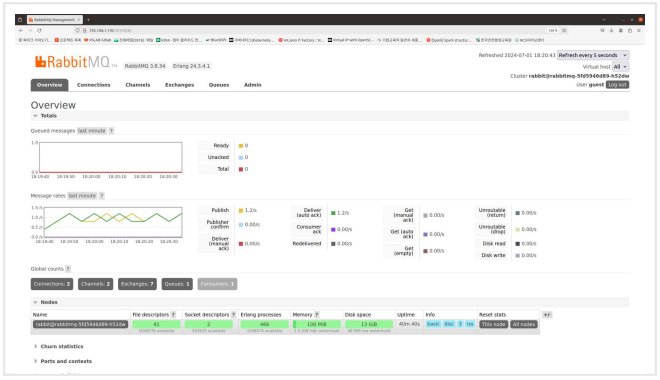


5.3. 데이터 전송확인

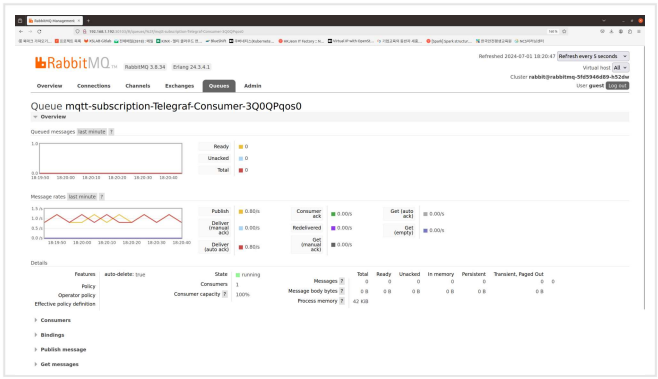
- 센서데이터를 전송합니다.

```
python3 random_data_push.py
```

- RabbitMQ의 웹에 접속해 데이터가 정상적으로 전송이 되는지 확인합니다.
  - IP : http://192.168.1.192:30103/
  - ID : guest
  - P/W : guest



새 글쓰기



- 마스터 노드인 `VM2`에 접속합니다.
  - 센서 데이터가 엣지서버를 거쳐 클라우드로 들어오는 데이터가 데이터베이스에 저장되는 지 확인합니다.

```
kubectl get all
```

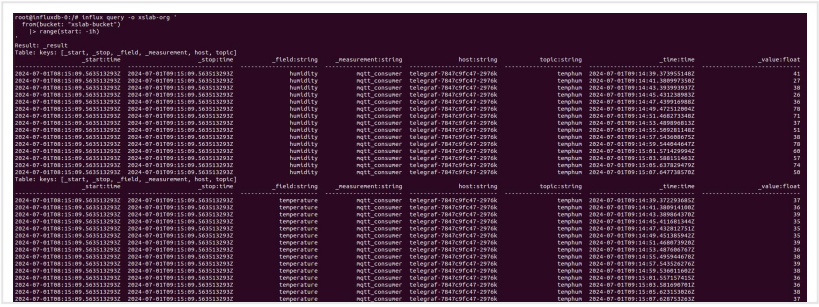
  - 데이터 전송 확인을 확인합니다. - Influx DB (master 노드에서 진행)

```
# Influx DB의 이름 확인
# kubectl exec -it <Influx DB 파드의 이름> bash
kubectl exec -it influxdb-0 bash

# influx에 로그인합니다.
influx config ls

influx config create \
  --config-name default \
  --host-url http://localhost:8086 \
  --org xslab-org \
  --token xslab-token \
  --active

influx config ls
```


```
## 데이터를 조회합니다.
influx query -o xslab-org '
  from(bucket: "xslab-bucket")
```

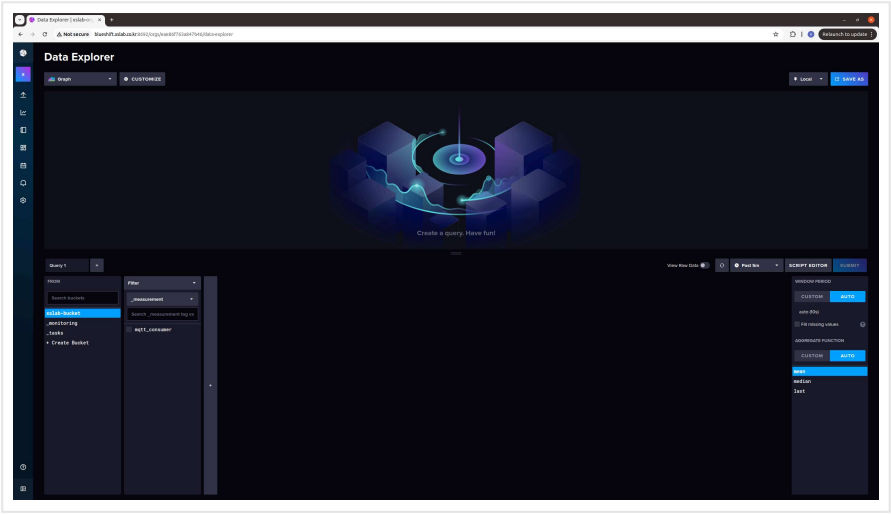
```
    |> range(start: -1h)
'
```

새 글쓰기



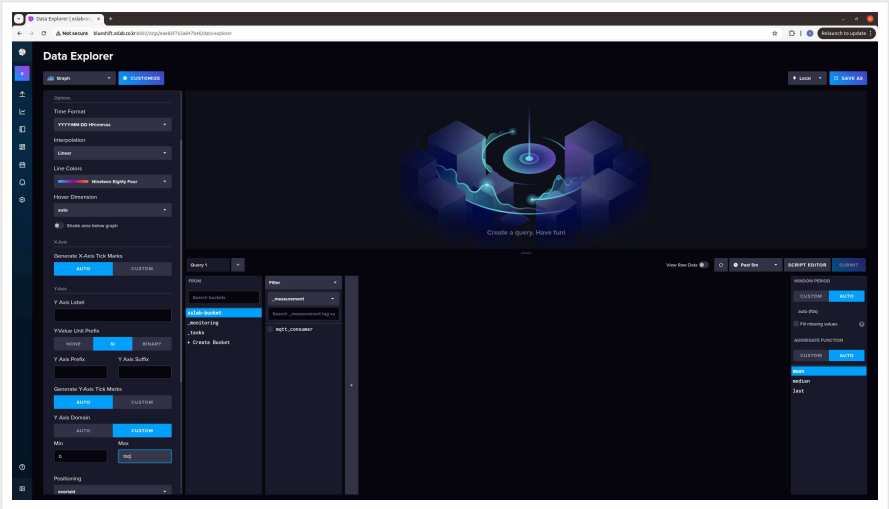
6. 데이터 시각화

- 6.1. Influx DB Web에 접속합니다.
  - IP : https://192.168.1.192:30201
  - ID : xslab
  - P/W : xslab1234
- 6.2. 왼쪽 탭에서 Data Explorer 선택합니다.

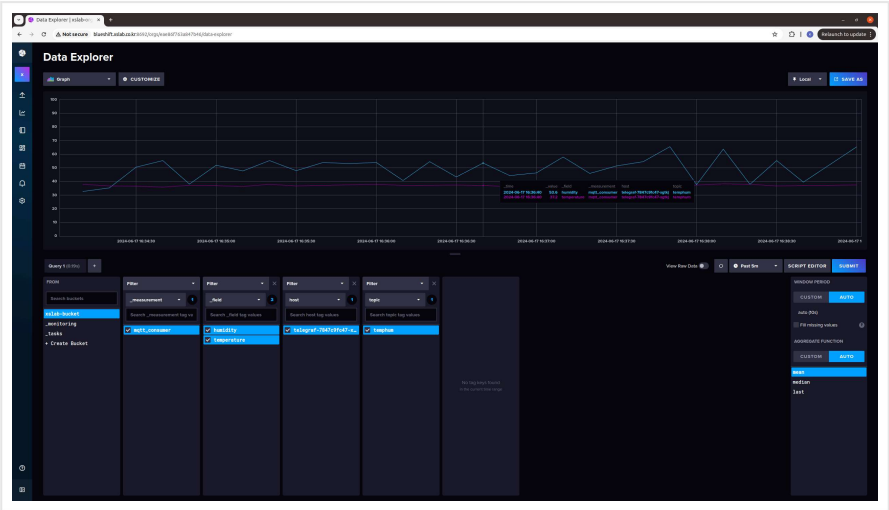


- 6.3. 좌측 상단의 CUSTOMIZE 선택 후 Y Axis Domain을 Custom으로 선택 후 최소값:0 최대값:100 으로 설정합니다.

새 글쓰기



- 6.4. CUSTOMIZE를 다시 선택해서 닫은 후 하단의 사진과 같이 시각화를 원하는 데이터 선택 후 우측의 submit 선택합니다.



Screenshot from 2024-07-01 11-33-19.png (64.85 Kb)
sensorClusterDeviceProfile.yaml (1.18 Kb)
Screenshot from 2024-07-01 14-03-59.png (153.54 Kb)
Screenshot from 2024-07-01 14-02-52.png (157.34 Kb)
Screenshot from 2024-07-01 14-13-10.png (146.74 Kb)
Screenshot from 2024-07-01 14-34-12.png (205.99 Kb)
Screenshot from 2024-07-01 14-36-10.png (229.64 Kb)
sensorClusterDeviceProfile.yaml (1.18 Kb)
Screenshot from 2024-07-01 14-38-08.png (306.41 Kb)
Screenshot from 2024-07-01 14-38-18.png (283.33 Kb)
Screenshot from 2024-07-01 14-38-39.png (282.32 Kb)

새 글쓰기

Screenshot from 2024-07-01 14-39-20.png (308.86 Kb)
Screenshot from 2024-07-01 14-39-29.png (282.78 Kb)
Screenshot from 2024-07-01 14-45-44.png (207.81 Kb)
Screenshot from 2024-07-01 14-45-53.png (222.23 Kb)
Screenshot from 2024-07-01 14-46-01.png (283.69 Kb)
Screenshot from 2024-07-01 14-48-39.png (26.32 Kb)
Screenshot from 2024-07-01 14-48-51.png (223.34 Kb)
Screenshot from 2024-07-01 14-49-09.png (333.78 Kb)
Screenshot from 2024-07-01 15-08-33.png (155.99 Kb)
Screenshot from 2024-07-01 15-07-13.png (158.98 Kb)
Screenshot from 2024-07-01 15-10-54.png (148.04 Kb)
Screenshot from 2024-07-01 15-01-56.png (224.04 Kb)
Screenshot from 2024-07-01 15-06-09.png (243.05 Kb)
Edge-X_3.1.postman_environment.json (443 bytes)
Screenshot from 2024-07-01 15-41-05.png (148.54 Kb)
Screenshot from 2024-07-01 16-29-34.png (168.24 Kb)
Screenshot from 2024-07-01 16-24-07.png (130.47 Kb)
Screenshot from 2024-07-01 16-33-32.png (111.31 Kb)
Screenshot from 2024-07-01 16-33-44.png (139.07 Kb)
Screenshot from 2024-07-01 16-33-54.png (225.23 Kb)
Screenshot from 2024-07-01 16-36-40.png (225.58 Kb)
Screenshot from 2024-07-01 16-37-01.png (69.40 Kb)
Screenshot from 2024-07-01 14-36-10.png (229.64 Kb)
Screenshot from 2024-07-01 17-12-44.png (133.60 Kb)
Screenshot from 2024-07-01 17-17-41.png (146.46 Kb)
Screenshot from 2024-07-01 17-17-14.png (22.31 Kb)
Screenshot from 2024-07-01 17-36-23.png (17.82 Kb)
Screenshot from 2024-07-01 17-40-07.png (64.02 Kb)
Screenshot from 2024-07-01 17-43-47.png (141.21 Kb)
Screenshot from 2024-07-01 17-45-59.png (209.11 Kb)
Screenshot from 2024-07-01 18-02-26.png (74.27 Kb)
Screenshot from 2024-07-01 18-15-24.png (270.55 Kb)

Screenshot from 2024-07-01 18-17-19.png (331.23 Kb)
Screenshot from 2024-07-01 18-17-34.png (325.06 Kb)
Screenshot from 2024-07-01 18-20-44.png (308.74 Kb)
Screenshot from 2024-07-01 18-20-50.png (310.61 Kb)
Edge-X_3.1.postman_collection.json (4.95 Kb)

새 글쓰기

글 지켜보기

댓글 0

편집 미리보기 체크리스트 추가

마크다운 도움말 Header Text Style Link List Checklist Image Blockquote Code Table Short Link

첨부할 파일을 끌어다 놓거나 파일 올리기 버튼을 클릭해서 선택하세요
클립보드 이미지를 붙여 넣을 수도 있습니다

댓글 입력

단축키 안내