MOS Cloud 기능동작 점검

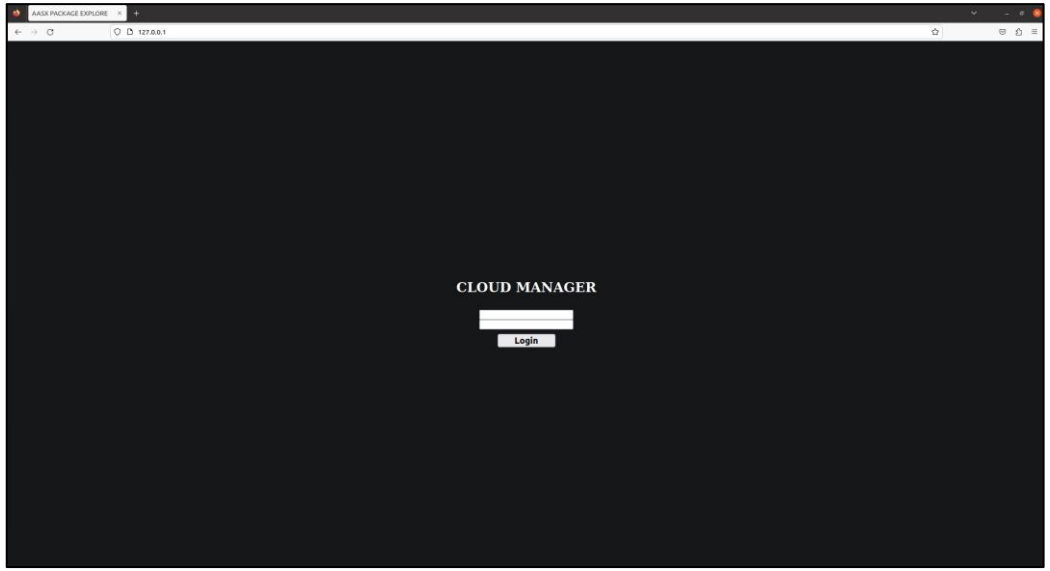
2023.08.03

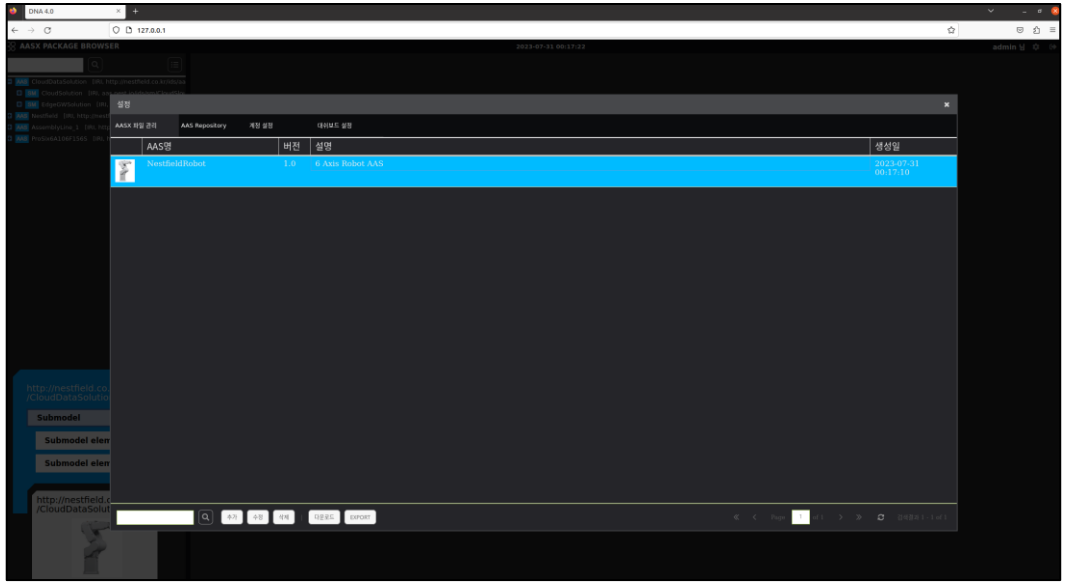
NEST + FIELD

네스트필드㈜

AASX Package Browser

- 웹 브라우저 실행 후 http://[MOS Cloud IP주소] 를 입력합니다.
- 로그인 계정은 설치 시 “python manage.py createsuperuser” 명령어로 입력한 계정 정보입니다.
- 우측 상단 설정버튼 – AAS 추가 기능을 이용하여 AAS 파일 업로드 및 조회 기능을 확인합니다.





OPCUA Data Acquisition (1)

- OPCUA 데이터 수집/저장 확인을 위해선 MOS Edge 설치 및 기능동작 점검이 선행되어야합니다.
- Github MOS Cloud 레포지토리의 Verification 폴더 내 파일을 /opt/cfg/ 에 위치시킵니다.
  - cd /opt/cfg
  - wget https://github.com/auto-mos/MOS-Packages/raw/main/Tiny%20Package/MOS%20Cloud/Verification/engineering.csv
  - wget https://github.com/auto-mos/MOS-Packages/raw/main/Tiny%20Package/MOS%20Cloud/Verification/nodeset.xml
  - wget https://github.com/auto-mos/MOS-Packages/raw/main/Tiny%20Package/MOS%20Cloud/Verification/syscfg.json
- syscfg.json 파일의 NetworkConnection 항목 IP 주소를 앞서 설치한 MOS Edge의 IP 주소로 변경합니다. (포트:4840)

OPCUA Data Acquisition (2)

- 모든 설정이 완료되면 sudo systemctl restart gather 명령어를 이용하여 OPCUA 데이터 수집 모듈을 재실행합니다.
- /opt/log/gather_날짜.log 파일 내용을 확인하여 데이터가 수집됨을 확인합니다.
- 정상적인 로그는 다음과 같습니다.

```
root@ubuntu:/opt/install# systemctl restart gather
root@ubuntu:/opt/install# tail -f ../log/gather_20230731.log
2023/07/31 00:46:44:083 [1] Waiting for OPN Response
2023/07/31 00:46:44:083 [1] Session disconnected
2023/07/31 00:46:44:105 [1] A SecureChannel to the server is open
2023/07/31 00:46:44:105 [1] Session disconnected
2023/07/31 00:46:44:105 [1] A SecureChannel to the server is open
2023/07/31 00:46:44:107 [1] A SecureChannel to the server is open
2023/07/31 00:46:44:109 [1] A SecureChannel to the server is open
2023/07/31 00:46:44:122 [1] A SecureChannel to the server is open
2023/07/31 00:46:44:122 [1] A session with the server is activated
2023/07/31 00:46:44:123 [1] Create subscription succeeded, id 1
2023/07/31 00:46:49:071 [1] gathering process (CPS) opcua = 30, amqp = 30
2023/07/31 00:46:54:072 [1] gathering process (CPS) opcua = 30, amqp = 30
2023/07/31 00:46:59:073 [1] gathering process (CPS) opcua = 30, amqp = 30
```

- 데이터 수집이 원활하지 않다면 /usr/lib/systemd/system/gather.service 파일의 내용을 확인합니다.
- OPCUA_NAME : 게이트웨이 이름 (default : MOS_GW)
  - AAS, syscfg.json, engineering.csv, MOS Edge의 gateway.config 파일과 설정내용이 같아야합니다.
- OPCUA_USER : OPCUA 계정 이름 (default : nestfield)
  - MOS Edge의 gateway.config 파일과 설정내용이 같아야합니다.
- OPCUA_PWD : OPCUA 계정 암호 (default : mos_opcua)
  - MOS Edge의 admin:~/sharedFolder/security/opcua.secured 파일과 설정내용이 같아야합니다.

```
[Unit]
Description=opcua gather service

[Service]
Environment="AMQP_USER=nestfield"
Environment="AMQP_PWD=ag13579!"
Environment="OPCUA_NAME=MOS_GW"
Environment="OPCUA_USER=nestfield"
Environment="OPCUA_PWD=mos_opcua"

Type=simple
ExecStart=/opt/bin/gather.sh
Restart=on-failure

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

NEST+ FIELD

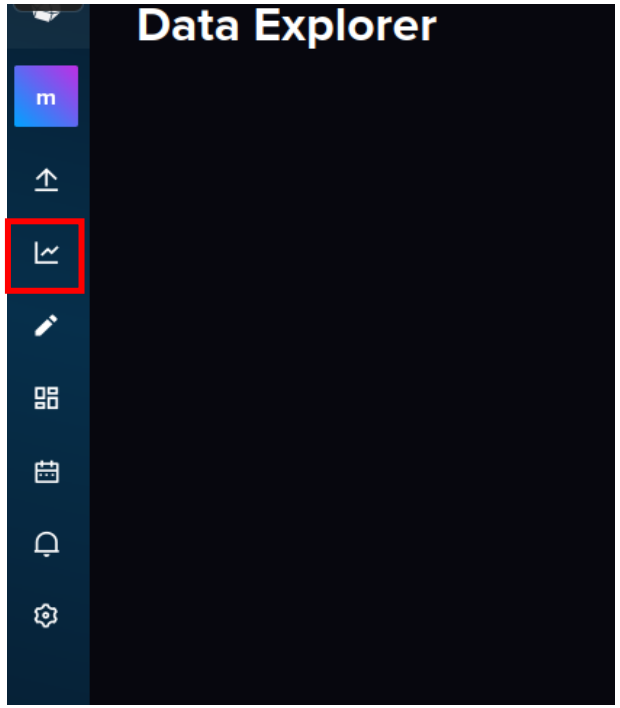
OPCUA Data Storage (1)

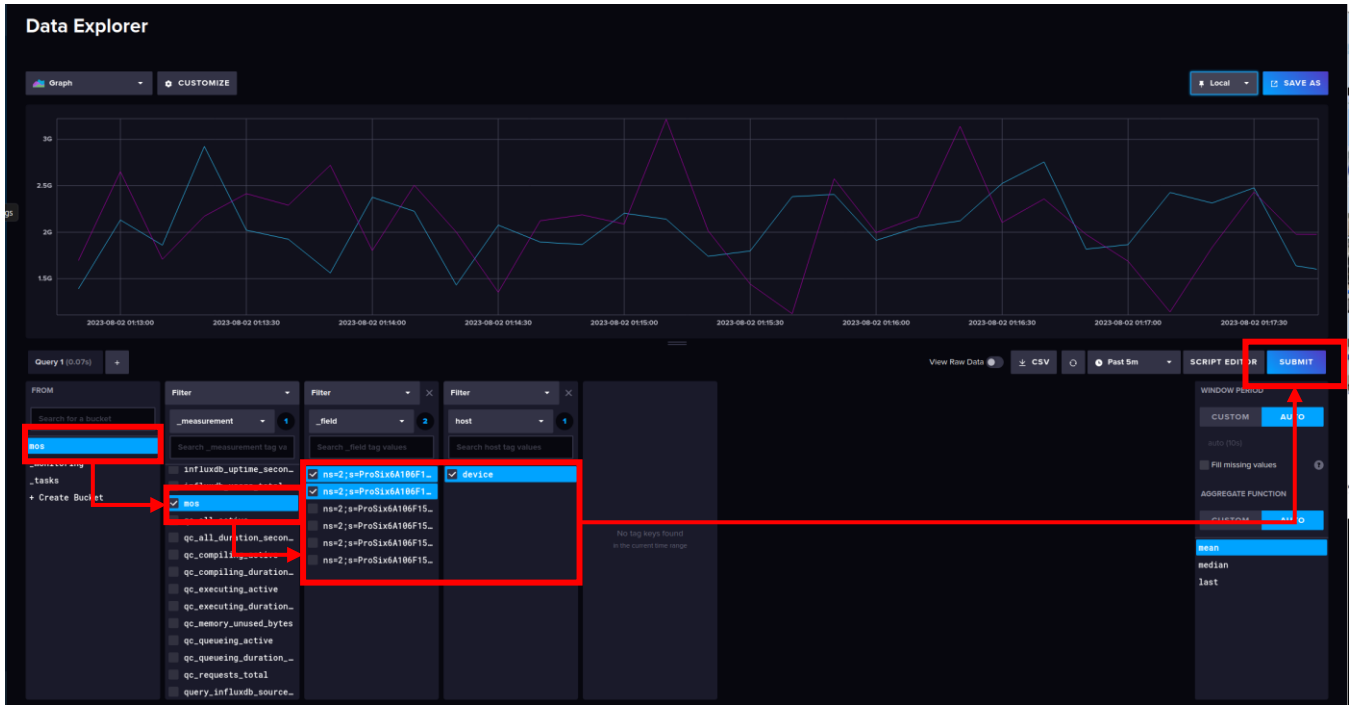
- Gather로 수집한 데이터가 InfluxDB에 정상적으로 저장되는 부분을 확인하는 방법입니다.
- systemctl restart itsdb 명령어를 이용하여 InfluxDB 저장 모듈을 재시작합니다.
- /opt/log/itsdb_날짜.log 파일 내용을 확인하여 다음과 같이 데이터가 저장됨을 확인합니다.

```
root@ubuntu:/opt/install# systemctl restart itsdb
root@ubuntu:/opt/install# tail -f /opt/log/itsdb_20230731.log
2023/07/31 00:57:42:971 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:57:47:971 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:57:52:972 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:57:54:850 [1] sigterm signal occured.
2023/07/31 00:57:54:850 [1] signal(15) captured
2023/07/31 00:57:54:850 [1] sigterm signal occured.
2023/07/31 00:57:54:958 [1] start itsdb program
2023/07/31 00:57:54:958 [1] success signal_monitor thread creation
2023/07/31 00:57:54:958 [1] success work_tsdb thread creation
2023/07/31 00:57:54:965 [1] amqp connected!!!
2023/07/31 00:57:59:959 [1] itsdb process (CPS) tsdb = 24
2023/07/31 00:58:04:959 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:58:09:960 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:58:14:961 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:58:19:961 [1] itsdb process (CPS) tsdb = 30
```

OPCUA Data Storage (2)

- 웹 브라우저 실행 후 http://[MOS Cloud 주소]:8086 을 입력하여 influxDB 웹 UI로 이동합니다.
- 좌측 'Data Explorer 버튼을 클릭한 후 아래 그림과 같이 데이터를 선택하여 우측 'SUBMIT' 버튼을 클릭합니다.
  - FROM : MOS / Filter – measurement – mos / 데이터태그 선택 후 시각화 데이터 확인





OPCUA Data Storage (3)

- 우측 그림처럼 itsdb process 숫자가 확인되지만 influxDB 웹UI에 데이터가 표시되지 않는 경우 아래 파일 내용을 확인합니다.
- /usr/lib/system/system/itsdb.service
  - Environment 하위 3개 항목(ORG/BUCKET/TOKEN) 내용 확인 후 서비스를 재시작합니다.
  - systemctl restart itsdb

```
[Unit]
Description=influxdb tsdb service

[Service]
Environment="AMQP_USER=app_itsdb"
Environment="AMQP_PWD=ait24680!"
Environment="TSDB_ORG=mos"
Environment="TSDB_BUCKET=mos"
Environment="TSDB_TOKEN=frsZuXqJLGN_-DydwTnycMyL8asQHLAPjx7B-aXm5m3IOUR7ctT_nI8SXo0dhcKyFDXVz_HjoadQDC7bRRDzkw=="

Type=simple
ExecStart=/opt/bin/itsdb.sh
Restart=on-failure
```

```
root@ubuntu:/opt/install# systemctl restart itsdb
root@ubuntu:/opt/install# tail -f /opt/log/itsdb_20230731.log
2023/07/31 00:57:42:971 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:57:47:971 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:57:52:972 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:57:54:850 [1] sigterm signal occured.
2023/07/31 00:57:54:850 [1] signal(15) captured
2023/07/31 00:57:54:850 [1] sigterm signal occured.
2023/07/31 00:57:54:958 [1] start itsdb program
2023/07/31 00:57:54:958 [1] success signal_monitor thread creation
2023/07/31 00:57:54:958 [1] success work_tsdb thread creation
2023/07/31 00:57:54:965 [1] amqp connected!!!
2023/07/31 00:57:59:959 [1] itsdb process (CPS) tsdb = 24
2023/07/31 00:58:04:959 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:58:09:960 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:58:14:961 [1] itsdb process (CPS) tsdb = 30
2023/07/31 00:58:19:961 [1] itsdb process (CPS) tsdb = 30
```

NEST+ FIELD

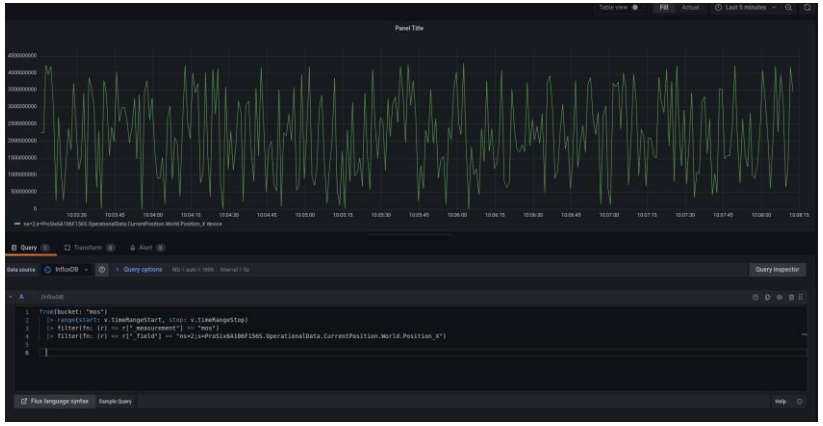
2D Dashboard

- influxDB의 Data Explorer에 데이터가 표시된다면 해당 데이터는 Grafana로 시각화할 수 있습니다.
- MOS Cloud Manual의 Setings – 기기 모니터링 데이터 대시보드 생성 항목을 참고하여 작성합니다.
- DB 쿼리문은 아래 우측 그림에 있는 내용대로 작성하면 그래프 조회가 가능합니다.

[점검용 DB 쿼리문]

```
from(bucket: "mos")
|> range(start: v.timeRangeStart, stop: v.timeRangeStop)
|> filter(fn: (r) => r["_measurement"] == "mos")
|> filter(fn: (r) => r["_field"] ==
"ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_X")
```

▪그래프 출력 예시



AMQP 메시징버스 이용 데이터 접근

- 아래 명령어를 통해 메시징버스 데이터 접근 스크립트를 다운로드합니다.
  - wget https://github.com/auto-mos/MOS-Packages/raw/main/Tiny%20Package/MOS%20Cloud/Verification/amqp_realtime.py
- 아래 명령어를 통해 데이터 접근용 AMQP 계정을 생성합니다.
  - rabbitmqctl add_user amqp_test amqp_test
  - rabbitmqctl set_user_tags amqp_test management
  - rabbitmqctl set_permissions -p / amqp_test ".*" ".*" ".*"
- 스크립트 실행을 위해 필요한 패키지를 설치합니다.
  - python –m pip install pika
- 스크립트를 실행합니다.
  - python ./amqp_realtime.py
- 데이터에 정상적으로 접근되는것을 확인할 수 있습니다.

```
Waiting for messages. To exit press CTRL+C
time: 1692669280559352000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_X, value: 964124096.000000
time: 1692669280559376000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_Y, value: 2965067520.000000
time: 1692669280559383000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_Z, value: 3751735040.000000
time: 1692669280559387000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_U, value: 3318233088.000000
time: 1692669280559391000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_V, value: 3408947712.000000
time: 1692669280559395000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_W, value: 3116576256.000000
time: 1692669281543551000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_X, value: 3036817152.000000
time: 1692669281543571000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_Y, value: 4275155200.000000
time: 1692669281543576000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_Z, value: 1654912640.000000
time: 1692669281543580000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_U, value: 3486371840.000000
time: 1692669281543583000, tag: ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_V, value: 1456934400.000000
```

- 스크립트의 아래 표시된 부분을 수정하여 다른계정을 사용하여 접근 또는 외부 환경에서도 접근할 수 있습니다.

```
credentials = pika.PlainCredentials(username='amqp_test', password='amqp_test')

connection = pika.BlockingConnection(pika.ConnectionParameters(host='127.0.0.1', port=5672, credentials=credentials))
channel = connection.channel()
```

NEST+ FIELD

REST API 이용 데이터 접근

- 아래 명령어를 통해 REST API 예제 스크립트를 다운로드합니다.
  - wget https://github.com/auto-mos/MOS-Packages/raw/main/Tiny%20Package/MOS%20Cloud/Verification/rest_lastdata.sh
  - wget https://github.com/auto-mos/MOS-Packages/raw/main/Tiny%20Package/MOS%20Cloud/Verification/rest_timerange.sh
- curl 패키지가 설치되어있지 않다면 아래 명령어를 통해 설치합니다.
  - sudo apt-get install curl
- 아래 명령어를 통해 스크립트를 실행 가능하도록 변경합니다.
  - chmod +x ./*.sh
- 스크립트를 열어 아래 내용을 변경합니다.
  - rest_lastdata.sh

```
curl --request POST \
http://127.0.0.1:8086/api/v2/query?org=mos \
--header 'Authorization: Token frsZuXqJLGN_-DydwTnycMyL8asQHLAPjx7B-aXm5m3IOUR7ctT_nI8SXo0dhcKyFDXVz_HjoadQDC7bRRDzkw==' \
--header 'Accept: application/csv' \
--header 'Content-type: application/vnd.flux' \
--data 'from(bucket: "mos")
  |> range(start: 0)
  |> filter(fn: (r) => r["_field"] == "ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_X")
  |> last()'
```

* org 및 Token 변경 (Django 토큰)

- rest_timerange.sh ※ 데이터 조회 시간 변경 시, 실제 한국 시간보다 9시간 느리게 설정하여야 합니다. (ex. 한국시간 11시 → 변경 시 2시, InfluxDB에서 기본 UTC 사용하기 때문)

```
curl --request POST \
http://127.0.0.1:8086/api/v2/query?org=mos \
--header 'Authorization: Token frsZuXqJLGN_-DydwTnycMyL8asQHLAPjx7B-aXm5m3IOUR7ctT_nI8SXo0dhcKyFDXVz_HjoadQDC7bRRDzkw==' \
--header 'Accept: application/csv' \
--header 'Content-type: application/vnd.flux' \
--data 'from(bucket: "mos")
  |> range(start: 2023-08-22T00:00:00.000Z, stop: 2023-08-22T23:59:59.000Z)
  |> filter(fn: (r) => r["_field"] == "ns=2;s=ProSix6A106F156S.OperationalData.CurrentPosition.World.Position_X")'
```

* org 및 Token 변경 (Django 토큰)

* 데이터 조회 시간 변경

NEST+ FIELD

REST API 이용 데이터 접근

- rest_lastdata.sh 실행 (설정한 태그의 마지막 값 조회)

데이터 저장 시점

데이터 값

데이터 이름

```
root@ubuntu:~# ./rest_lastdata.sh
,result,table,_start,_stop,_time,_value,_field,_measurement,host
,_result,0,1970-01-01T00:00:00Z,2023-08-22T02:17:28.440639041Z,2023-08-22T02:17:26.487073Z,4030755840,ns=2;s=ProSix6A106F156S.Operat
ionalData.CurrentPosition.World.Position_X,mos,device
```

- rest_timerange.sh 실행 (설정한 태그의 입력 기간동안 데이터 조회)

```
,result,table,_start,_stop,_time,_value,_field,_measurement,host
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:01.667514Z,4176968704,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:02.657405Z,1831092480,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:03.656975Z,1211302400,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:04.659312Z,724568256,ns=2;s=ProSix6A106F156S.OperationalData.C
urrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:05.658608Z,2306082816,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:06.658736Z,2143555712,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:07.656647Z,831754688,ns=2;s=ProSix6A106F156S.OperationalData.C
urrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:08.657249Z,3748635904,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:09.657602Z,4259549440,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:10.658412Z,1051731008,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:11.65946Z,512554464,ns=2;s=ProSix6A106F156S.OperationalData.Cu
rrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:12.658807Z,2538932992,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:13.657403Z,1523815680,ns=2;s=ProSix6A106F156S.OperationalData.
CurrentPosition.World.Position_X,mos,device
,_result,0,2023-08-22T00:00:00Z,2023-08-22T23:59:59Z,2023-08-22T00:19:14.65834Z,2723760640,ns=2;s=ProSix6A106F156S.OperationalData.C
urrentPosition.World.Position_X,mos,device
```

- 스크립트 내 IP주소 변경 시 외부에서도 접근할 수 있습니다.

NEST+ FIELD