AUTOMATED FRAMEWORK FOR GENERATING CYBER-PHYSICAL RANGE FOR SMART GRID

USER MANUAL
FOR SMART GRID CYBER RANGE

ADSC, NUS

June 25, 2023

Contents

1 Introduction

This document provides a guide to generate a Smart Grid Cyber Range (SGCR) based on user-defined models. The setup is to be made on a Linux operating system. In this example, Ubuntu 20.04 LTS is used.

1.1 OS and Python dependencies required

The following are the tools, utilities and libraries that will be used. Details on the installation can be found in the respective sections of each SGCR component. Refer to requirements.txt for the specific versions of Python dependencies.

OS dependencies

(1) gcc
(2) python3
(3) python3-pip
(4) mysql-server
(5) libiec61850
(6) xmllint
(7) nlohmann-json3-dev
(8) libmysql++-dev
(9) openjdk-8-jdk
(10) mininet
(11) openvswitch-testcontroller
(12) xterm
(13) jq
(14) openPLC61850
(15) ScadaBR
(16) rapid-xml
(17) wireshark (optional)

Python dependencies

(1) pandapower
(2) pymysql
(3) matplotlib
(4) python-igraph
(5) xmltodict
(6) lxml

2 Database

This is a one-time setup.

2.1 Dependencies required

Install MySQL (either on localhost or in another host) by referring to the official manual.

2.2 Steps to set up the database

(1) Create an empty database

(2) Restore the pandapower database structure with initial data using the given pandapower_db_initialization.sql file
sudo mysql -u [user] -p [database] < pandapower_db_initialization.sql

(3) Modify files <dir>/IED/db_config.txt and <dir>/Panda-db/DBTransmitter.py with the right **Username**, **Password** and **Database**

3 Virtual IED

3.1 Dependencies required

(1) libiec61850
run **make** in <dir>/IED/02_build/libs/libiec61850

(2) xmltodict
pip install xmltodict

(3) xmllint
sudo apt install libxml2-utils

(4) nlohmann-json3-dev
sudo apt install nlohmann-json3-dev

(5) libmysql++-dev
sudo apt install libmysql++-dev

(6) openjdk-8-jdk
sudo apt install openjdk-8-jdk

3.2 Folder contents

ICD files, CPMapping.xml, Thresholds.xml, db_config.txt, ied_build.sh, ied_getip.sh, ied_start.sh, xmllint and .libs, and folders 01_input, 02_build as listed in Fig. 1.

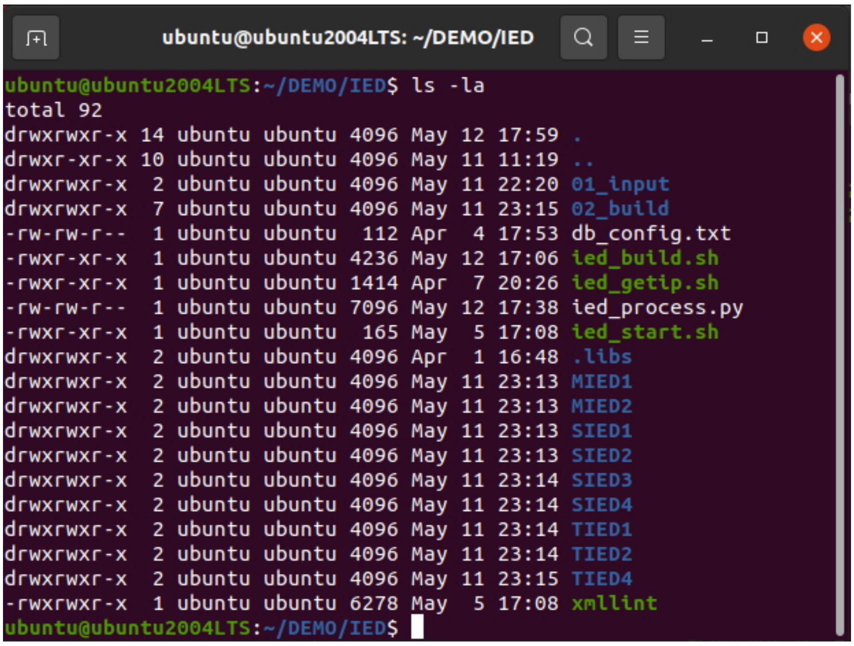

Figure 1

3.3 Building Virtual IEDs:

This is a one-time setup and will only be required to be repeated if there are changes in the CPMapping, Thresholds or ICD files.

(1) Required user input files below are placed in “01_input” folder:

- CPMapping.xml
- Thresholds.xml
- ICD files (<IED_name>.icd)

(2) Command to build:
cd <dir>/IED
./ied_build.sh

(3) The build will perform the following for each IED:

- Convert CPMapping.xml to <IED_name>_CPMapping.json
- Convert Thresholds.xml to <IED_name>_Threshold.json
- Build VIRTUAL_IED
- Place the respective files into folder <IED_name>, as shown in the example in Fig. 2.

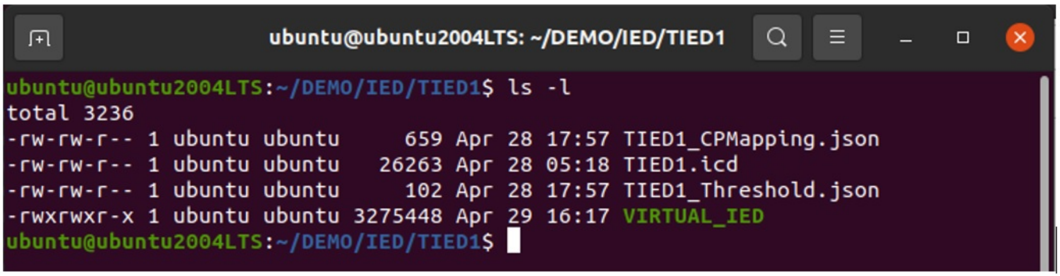

Figure 2

4 Cyber Network Simulation

4.1 Dependencies required

(1) Mininet

The installation instruction can be found in http://mininet.org/download/

(2) OpenVSwitch Testcontroller

sudo apt install openvswitch-testcontroller

sudo ln -s /usr/bin/ovs-testcontroller /usr/bin/controller

(3) xterm

sudo apt install xterm

(4) jq

sudo apt install jq

4.2 Commands to start cyber network simulation

Ensure the database and virtual IEDs are set up before this step.

(1) cd <dir>/Mininet

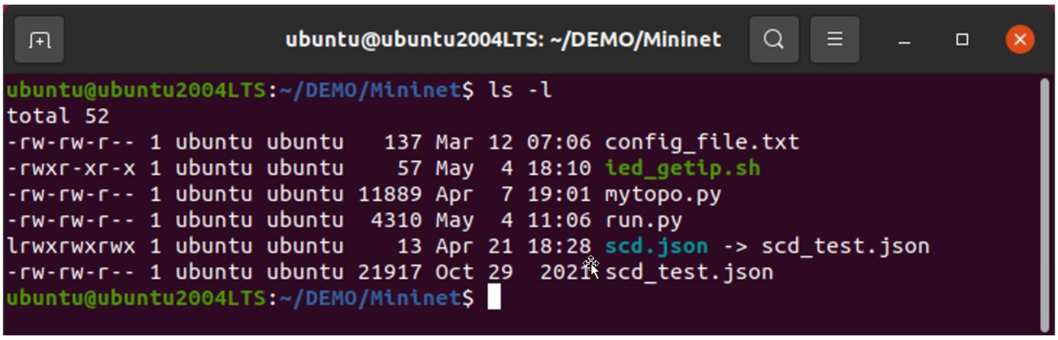

Figure 3

(2) Run command:

sudo python3 run.py

Output:

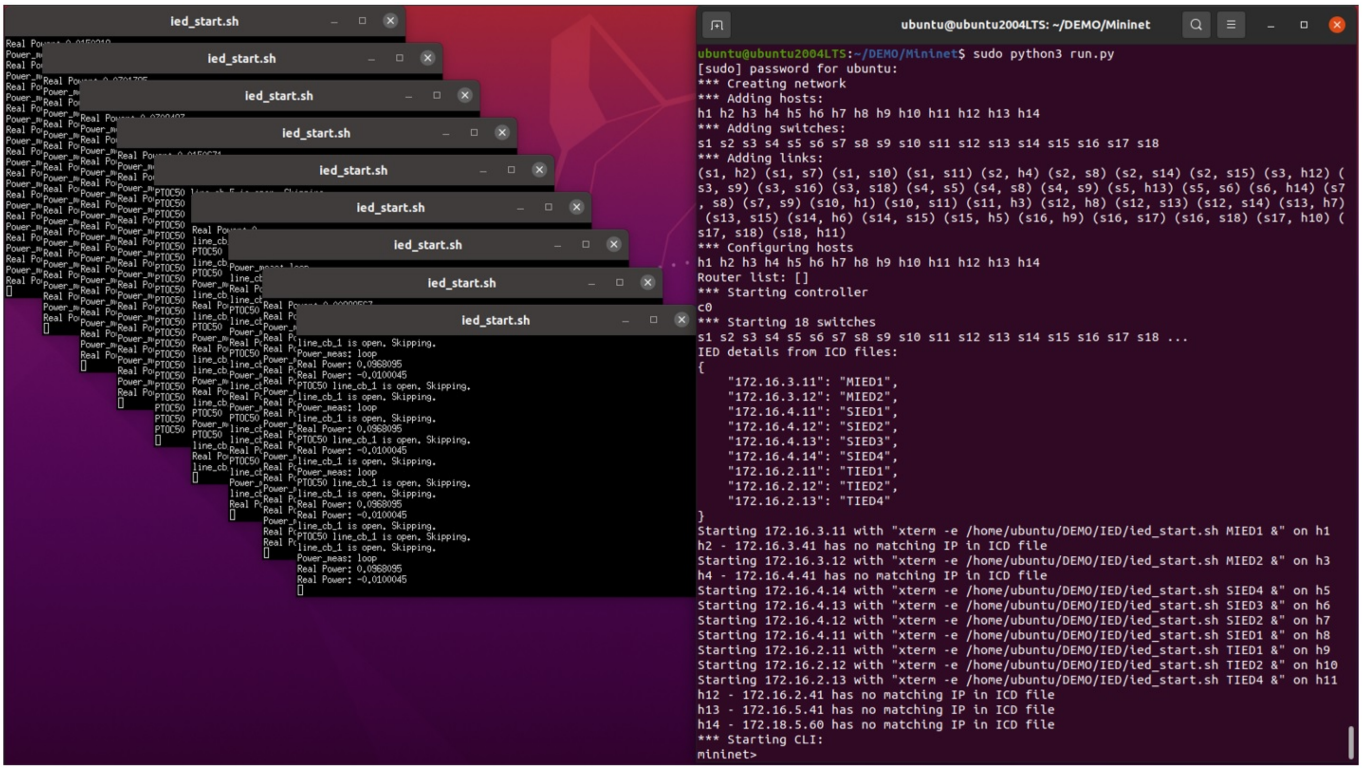
ied_start.sh
line_cb_1 is open. Skipping.
Power_meas: loop
Real Power: 0.0968095
Real Power: -0.0100045
PTOC50 line_cb_1 is open. Skipping.
line_cb_1 is open. Skipping.
Power_meas: loop
line_cb_1 is open. Skipping.
Real Power: 0.0968095
PTOC50 line_cb_1 is open. Skipping.
Real Power: -0.0100045
line_cb_1 is open. Skipping.
Power_meas: loop
PTOC50 line_cb_1 is open. Skipping.
line_cb_1 is open. Skipping.
Real Power: 0.0968095
Real Power: -0.0100045
line_cb_1 is open. Skipping.
PTOC50 line_cb_1 is open. Skipping.
line_cb_1 is open. Skipping.
Power_meas: loop
Real Power: 0.0968095
Real Power: -0.0100045
ubuntu@ubuntu2004LTS: ~/DEMO/Mininet
ubuntu@ubuntu2004LTS:~/DEMO/Mininet$ sudo python3 run.py
[sudo] password for ubuntu:
*** Creating network
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18
*** Adding links:
(s1, h2) (s1, s7) (s1, s10) (s1, s11) (s2, h4) (s2, s8) (s2, s14) (s2, s15) (s3, h12) (
s3, s9) (s3, s16) (s3, s18) (s4, s5) (s4, s8) (s4, s9) (s5, h13) (s5, s6) (s6, h14) (s7
, s8) (s7, s9) (s10, h1) (s10, s11) (s11, h3) (s12, h8) (s12, s13) (s12, s14) (s13, h7)
(s13, s15) (s14, h6) (s14, s15) (s15, h5) (s16, h9) (s16, s17) (s16, s18) (s17, h10) (
s17, s18) (s18, h11)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14
Router list: []
*** Starting controller
c0
*** Starting 18 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 ...
IED details from ICD files:
{
"172.16.3.11": "MIED1",
"172.16.3.12": "MIED2",
"172.16.4.11": "SIED1",
"172.16.4.12": "SIED2",
"172.16.4.13": "SIED3",
"172.16.4.14": "SIED4",
"172.16.2.11": "TIED1",
"172.16.2.12": "TIED2",
"172.16.2.13": "TIED4"
}
Starting 172.16.3.11 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh MIED1 &" on h1
h2 - 172.16.3.41 has no matching IP in ICD file
Starting 172.16.3.12 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh MIED2 &" on h3
h4 - 172.16.4.41 has no matching IP in ICD file
Starting 172.16.4.14 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh SIED4 &" on h5
Starting 172.16.4.13 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh SIED3 &" on h6
Starting 172.16.4.12 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh SIED2 &" on h7
Starting 172.16.4.11 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh SIED1 &" on h8
Starting 172.16.2.11 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh TIED1 &" on h9
Starting 172.16.2.12 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh TIED2 &" on h10
Starting 172.16.2.13 with "xterm -e /home/ubuntu/DEMO/IED/ied_start.sh TIED4 &" on h11
h12 - 172.16.2.41 has no matching IP in ICD file
h13 - 172.16.5.41 has no matching IP in ICD file
h14 - 172.18.5.60 has no matching IP in ICD file
*** Starting CLI:
mininet>

Figure 4

5 PLC

5.1 Dependencies required

(1) Install OpenPLC61850, by downloading from GitHub repository:
git clone https://github.com/smartgridadsc/OpenPLC61850.git
cd OpenPLC61850 && sudo ./install.sh linux

(2) Replace <dir>/OpenPLC61850/webserver/scl_client_files and scl_server_files with <dir>/examples/single_substation/OpenPLC61850/scl_client_files and scl_server_files respectively for the single substation example

5.2 Commands to start the OpenPLC:

(1) Open a mininet node

(2) From Mininet CLI, open a terminal in a PLC node, e.g. h13
mininet> xterm h13

(3) In the terminal, start OpenPLC:
cd <dir>/OpenPLC61850 && ./start_openplc.sh

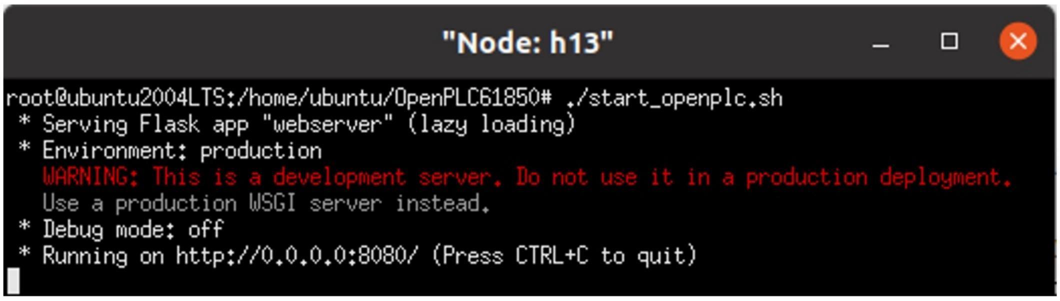

Figure 5

(4) In the same terminal, launch Firefox or Chrome browser and access OpenPLC:
sudo -u <ubuntu_user> firefox & (or google-chrome if it is installed)
On browser, load http://localhost:8080 and log in as openplc/openplc

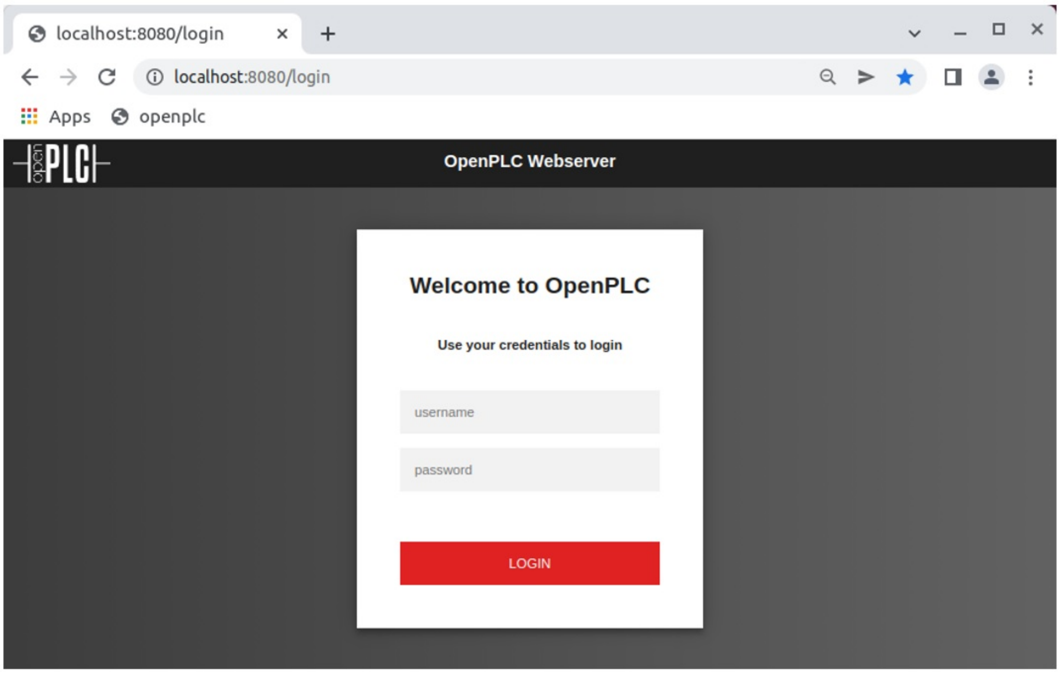
localhost:8080/login
localhost:8080/login
Apps
openplc
OpenPLC Webserver
Welcome to OpenPLC
Use your credentials to login
username
password
LOGIN

Figure 6

(5) Go to Program -> Choose File -> select the given .st file -> Upload Program (one-time setup)
For the single substation example, use
<dir>/examples/single_substation/OpenPLC61850/st_files/singlesub.st

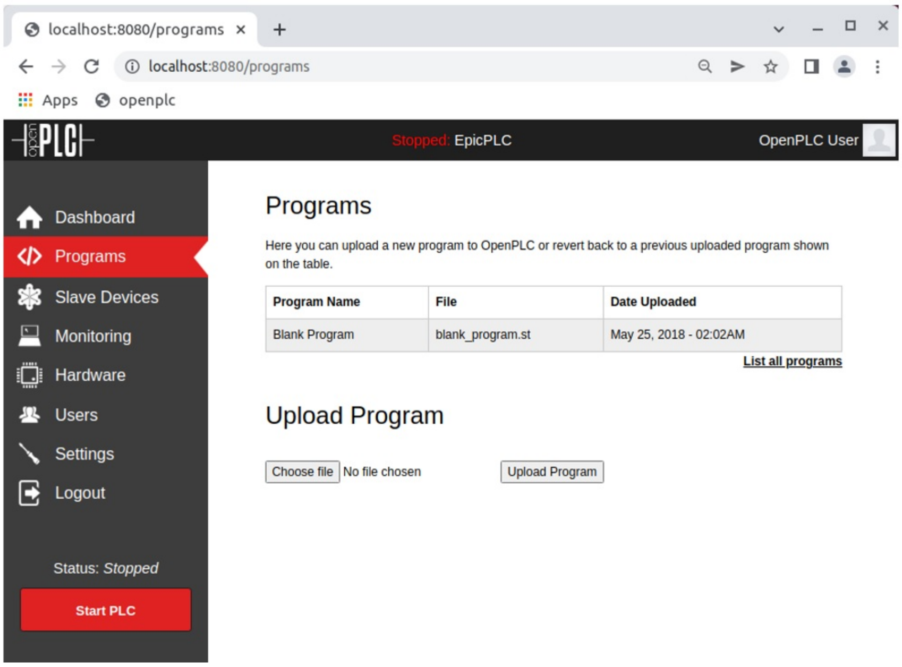

Figure 7

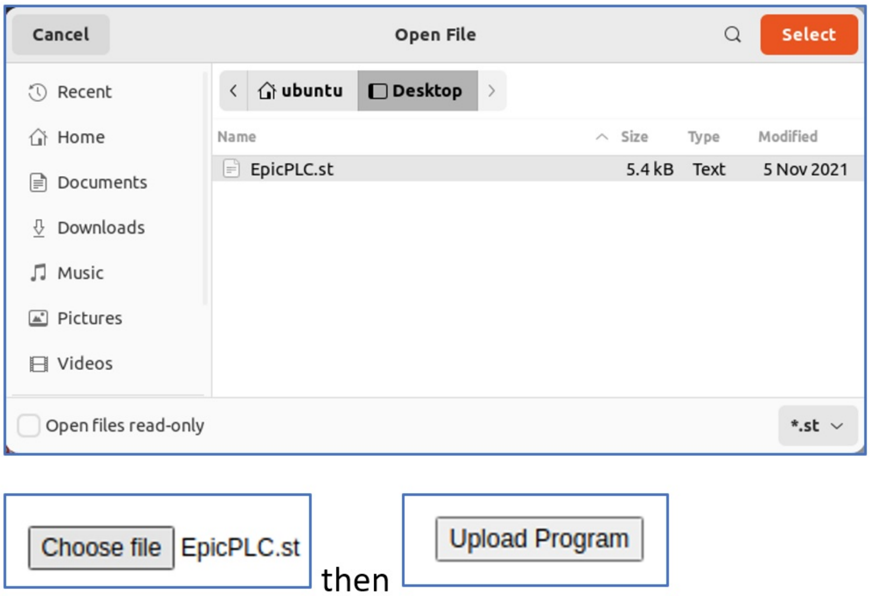

Figure 8

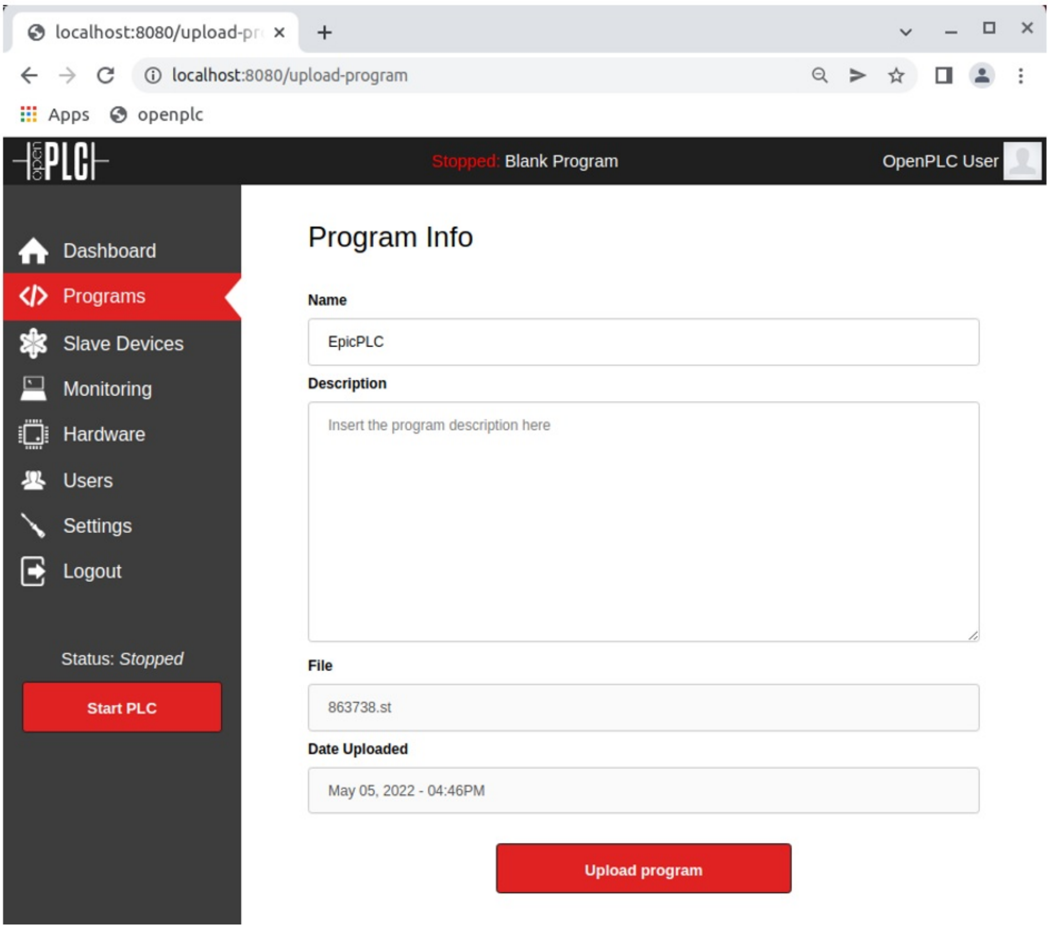

Figure 9

(6) In Program -> select the uploaded program -> Launch Program (to compile program)

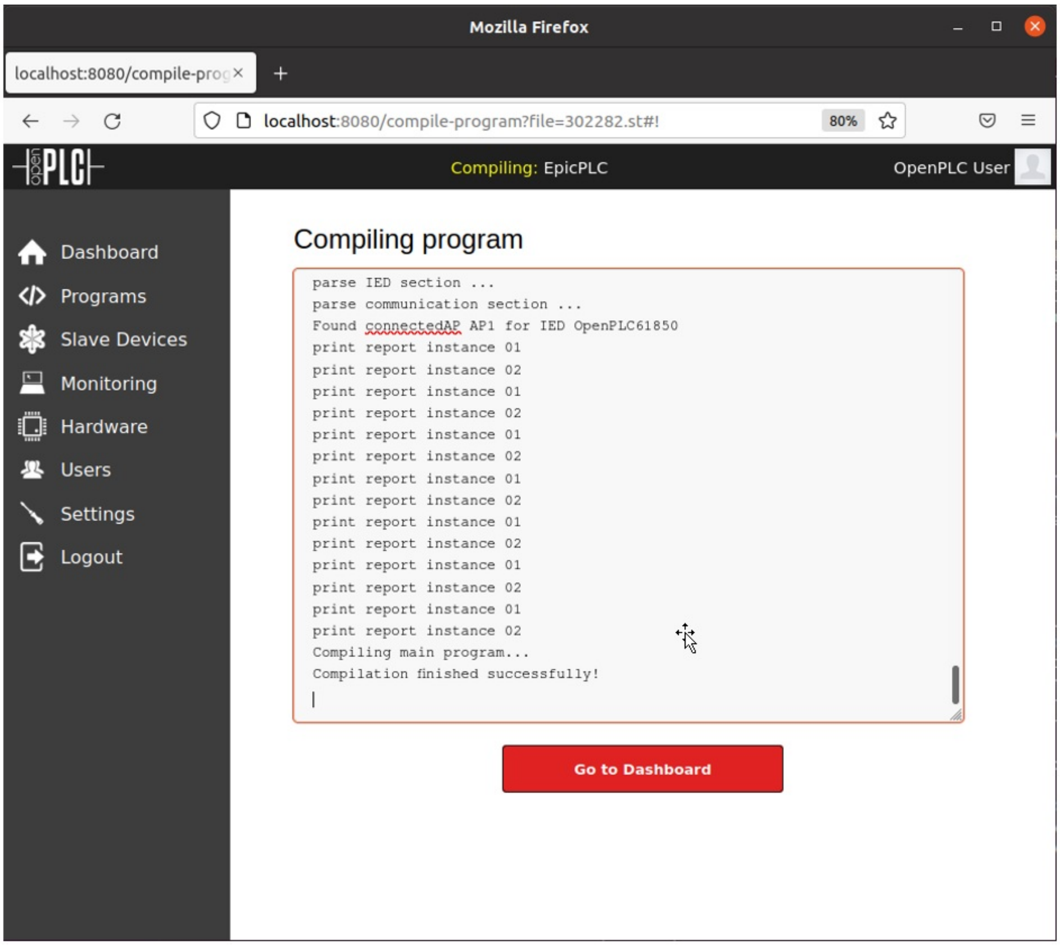

Figure 10

(7) Go to Dashboard -> Start PLC

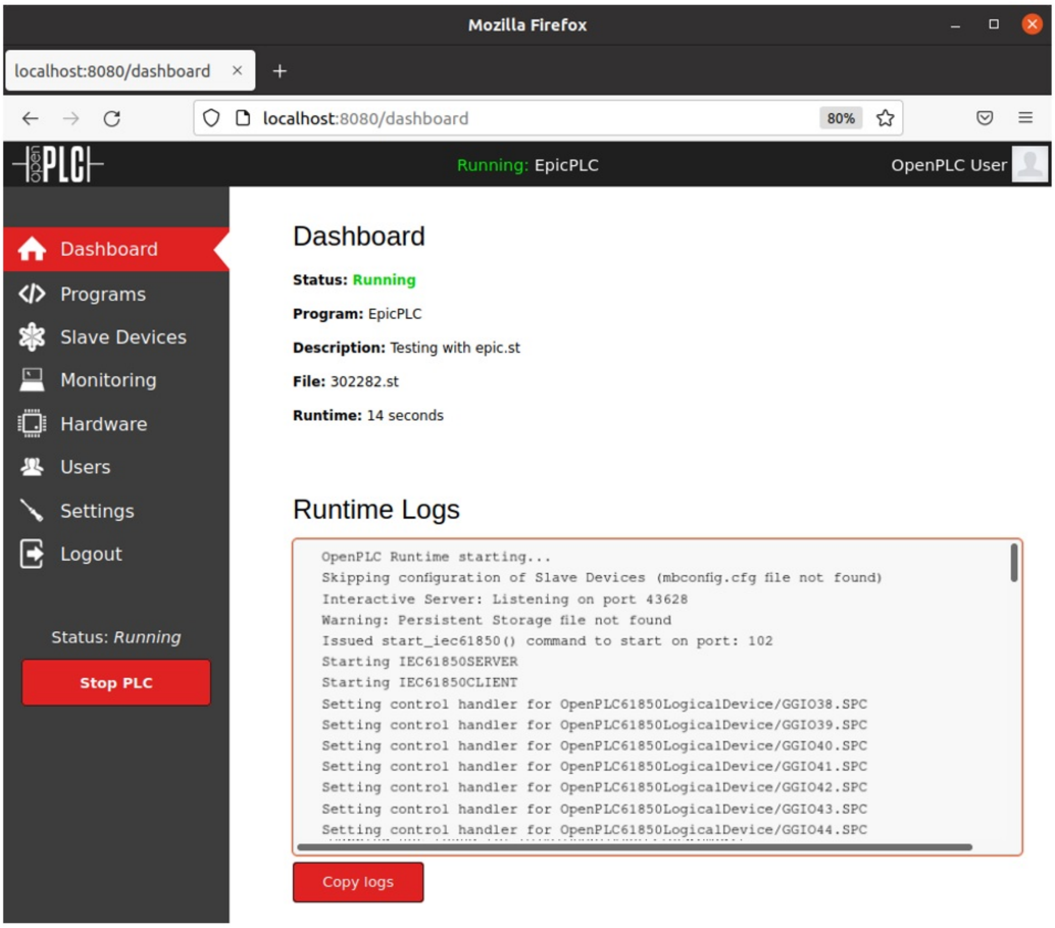

Figure 11

(8) Go to Monitoring (if no data is shown here, click Stop PLC and Start PLC again)

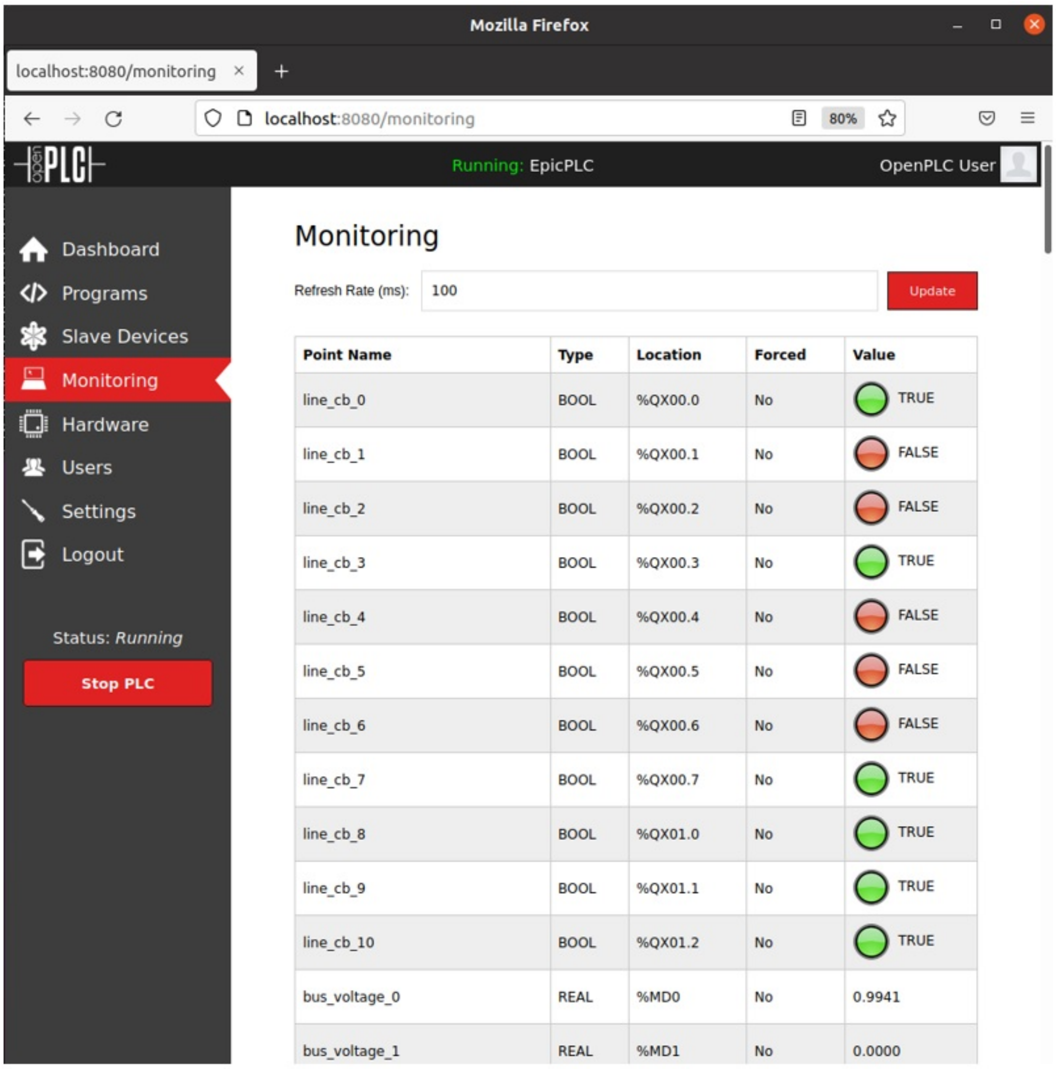
Mozilla Firefox
localhost:8080/monitoring
localhost:8080/monitoring
80%
Running: EpicPLC
OpenPLC User
Dashboard
Programs
Slave Devices
Monitoring
Hardware
Users
Settings
Logout
Status: Running
Stop PLC
Monitoring
Refresh Rate (ms):
100
Update
Point Name
Type
Location
Forced
Value
line_cb_0 BOOL %QX00.0 No TRUE
line_cb_1 BOOL %QX00.1 No FALSE
line_cb_2 BOOL %QX00.2 No FALSE
line_cb_3 BOOL %QX00.3 No TRUE
line_cb_4 BOOL %QX00.4 No FALSE
line_cb_5 BOOL %QX00.5 No FALSE
line_cb_6 BOOL %QX00.6 No FALSE
line_cb_7 BOOL %QX00.7 No TRUE
line_cb_8 BOOL %QX01.0 No TRUE
line_cb_9 BOOL %QX01.1 No TRUE
line_cb_10 BOOL %QX01.2 No TRUE
bus_voltage_0 REAL %MD0 No 0.9941
bus_voltage_1 REAL %MD1 No 0.0000

Figure 12

6 SCADA

6.1 Dependencies required

(1) Python3

(2) Python modules (for ScadaBR_Config):

- lxml
  pip install lxml
- xmltodict
  pip install xmltodict

(3) Download ScadaBR_Config from GitHub repository
(https://github.com/smartgridadsc/CyberRange/tree/main/single_substation/ScadaBR)

(4) Install ScadaBR, by downloading from GitHub:
git clone https://github.com/thiagoralves/ScadaBR_Installer.git
cd ScadaBR_Installer && sudo ./install_scadabr.sh

(5) Initial pandapower data must exist in the database (run python3 Simulator.py for 1 cycle to populate then Ctrl-C)

6.2 ScadaBR configuration

This script converts the ScadaBR configuration in XML to JSON. This is a one-time setup.

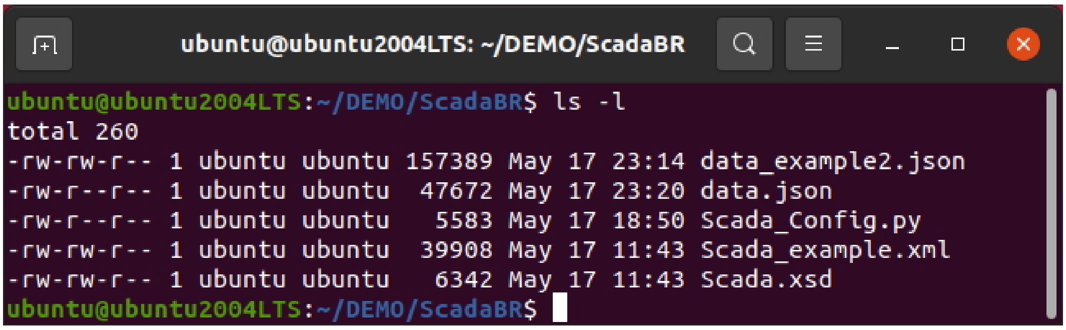

Figure 13

Command to run:
sudo python3 Scada_Config.py Scada_example.xml Scada.xsd
where Scada_example.xml is the input file and data.json is the converted output file.

Output:

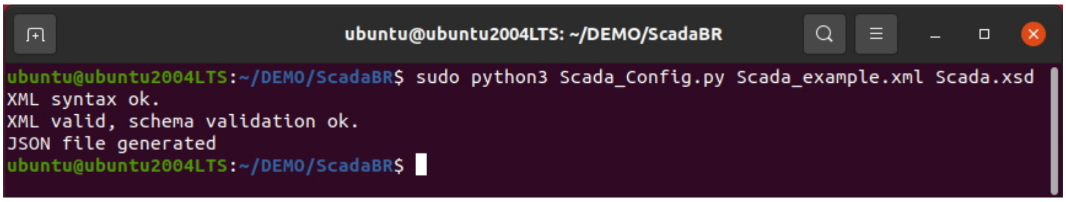

Figure 14

6.3 Commands to start the ScadaBR

(1) Open a mininet node (e.g. xterm h14)

(2) In the terminal, start ScadaBR:
/opt/tomcat6/apache-tomcat-6.0.53/bin/startup.sh

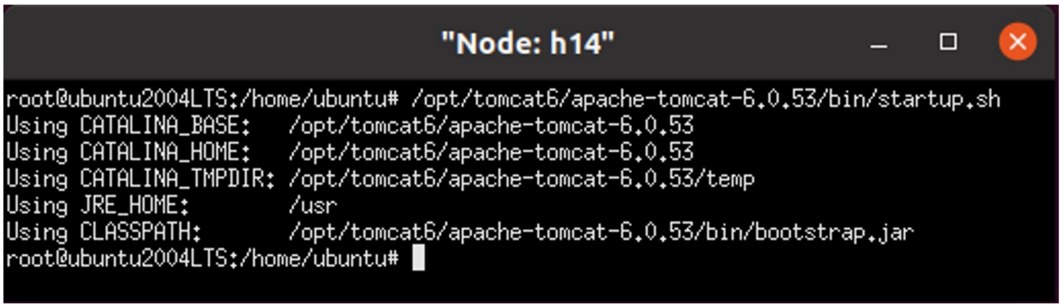

Figure 15

(3) Access ScadaBR on the web browser:
http://<ip_of_mininet_node_running_scadabr>:9090/ScadaBR/login.htm
and log in as admin/admin

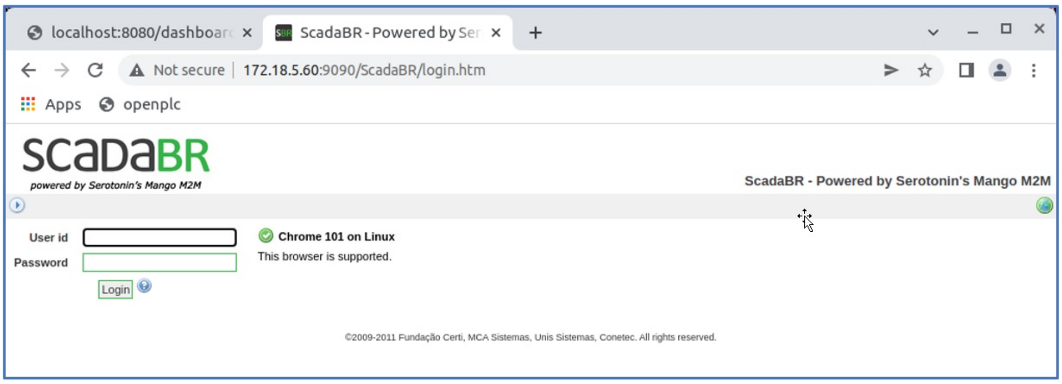

Figure 16

(4) Import data (one-time setup):
Select Import/Export icon in ScadaBR after logging in
Paste content of data.json (converted by Scada_Config.py script) and click Import

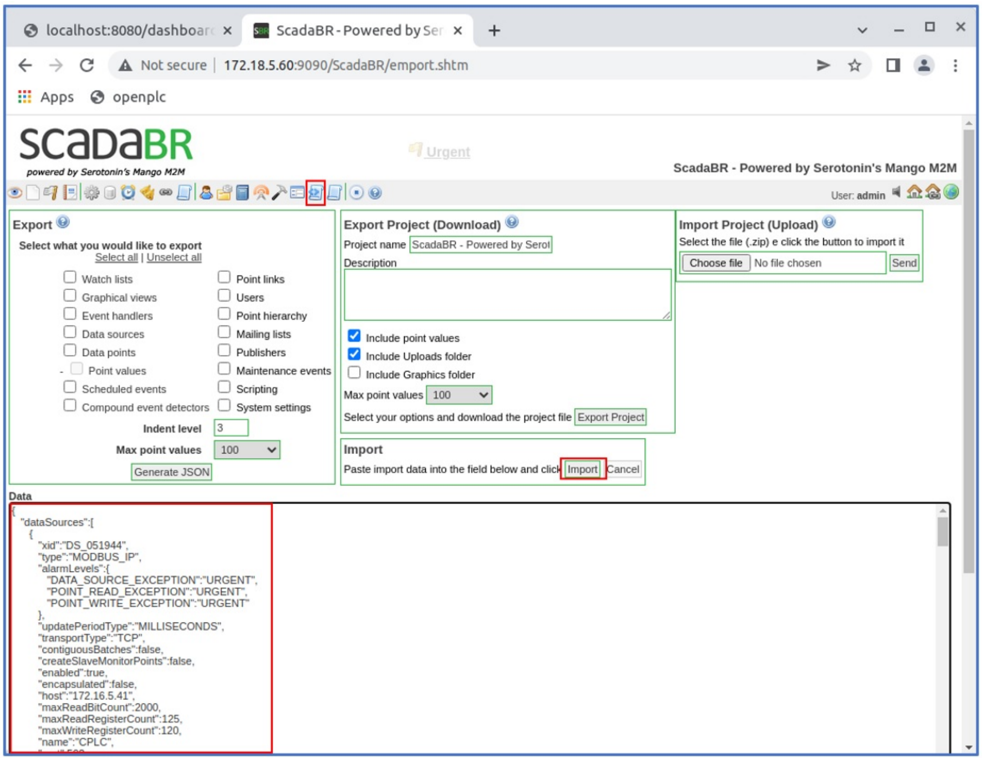

Figure 17

In ScadaBR, click Watch list icon to view the data metrics

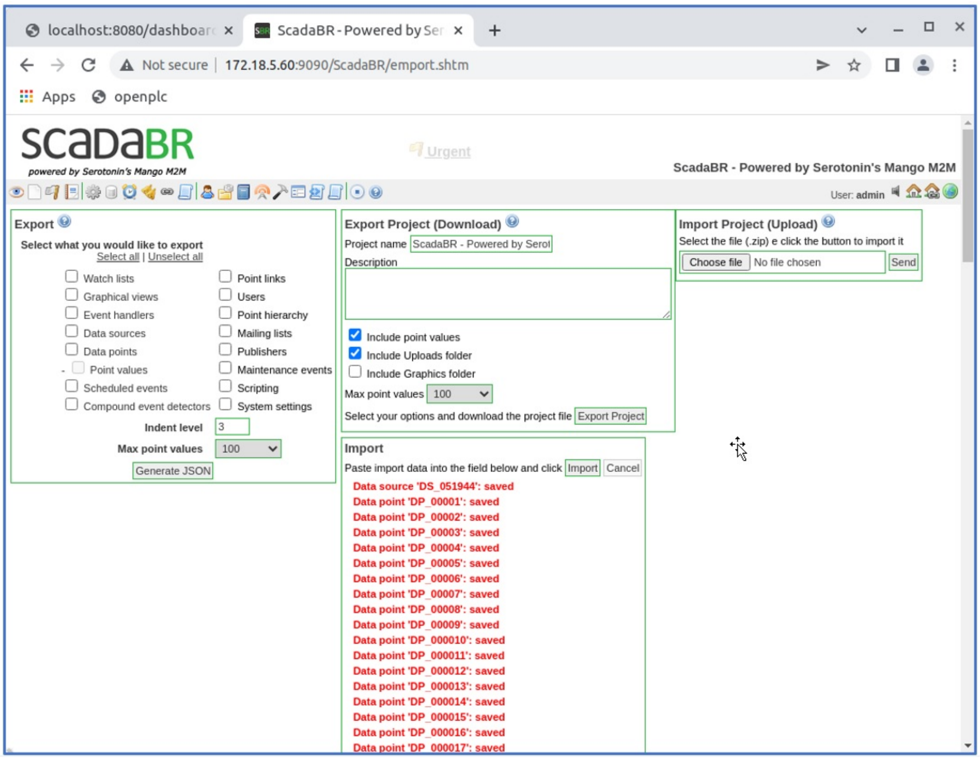

Figure 18

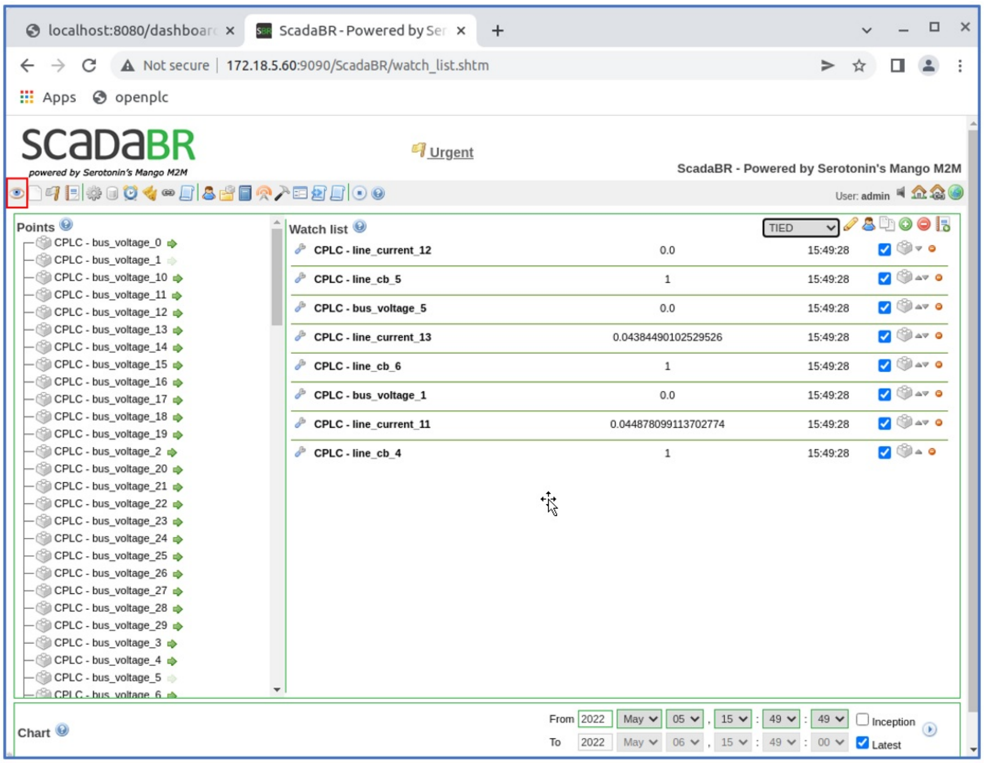

Figure 19

If required, click Data sources icon to enable/disable the data source and data points

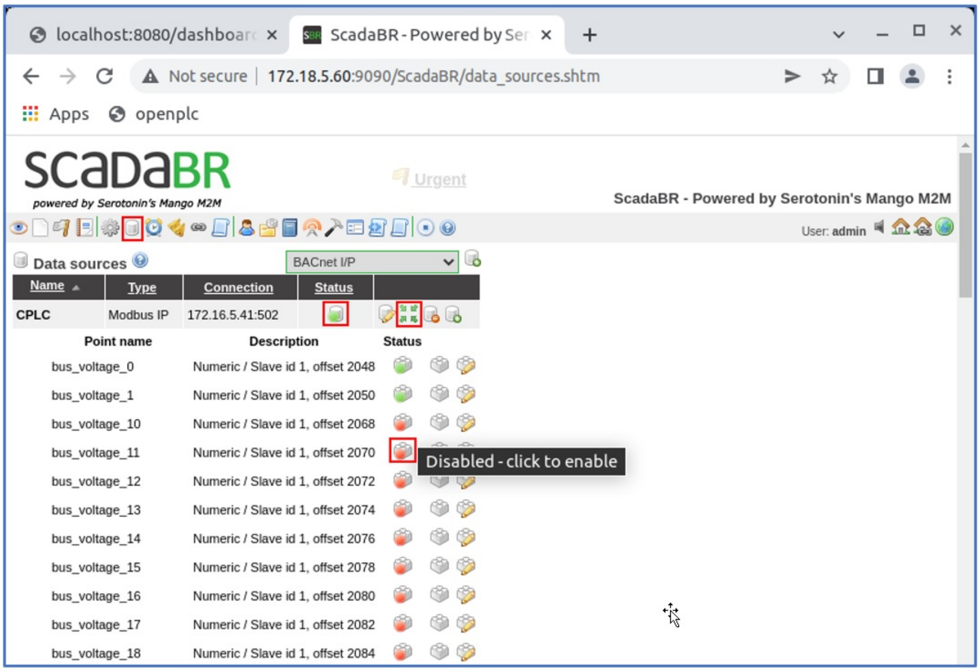

Figure 20

7 Pandapower Simulation

SSD file is input to create the SLD of the electrical system. The parameter specification detail are provided by extra configuration XML files.

7.1 Dependencies required

(1) Install pandapower, by running the command:
pip install pandapower
(http://www.pandapower.org/start/#install)

(2) Python3

(3) Python modules:

- pymysql
  pip install pymysql
- matplotlib
  pip install matplotlib
- python-igraph
  pip install python-igraph

7.2 Folder contents

(1) Standalone system
SSD file, extra_config.xml, extra_config_parser.py, process_ssd.py as listed in Fig. 21.

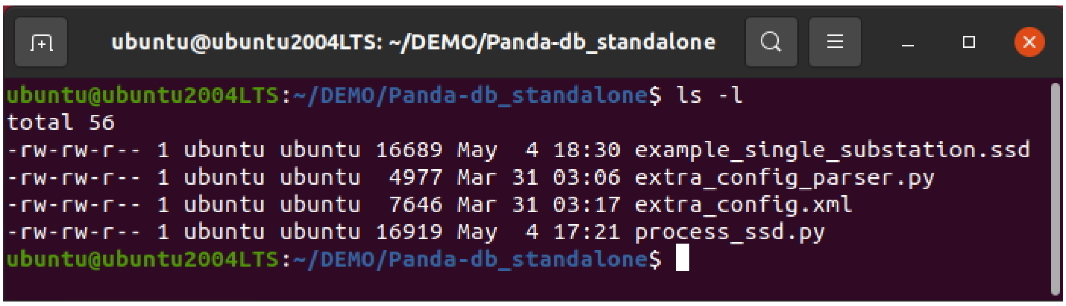

Figure 21

(2) Dynamic system (cyber network simulation with virtual IEDs, PLC and SCADA)
SSD file, extra_config.xml, extra_config_parser.py, process_ssd.py, Simulator.py, Constants.py, DBTransmitter.py, Logger.py, Network.py, running_status.txt and Logs folder as listed in Fig. 22.

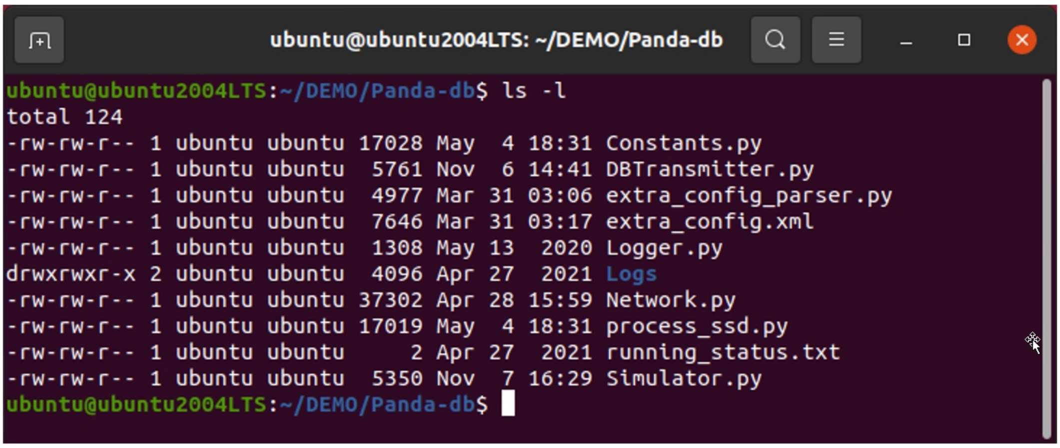

Figure 22

7.3 Commands to run the Pandapower simulation

(1) Standalone system

python3 process_ssd.py

Output:

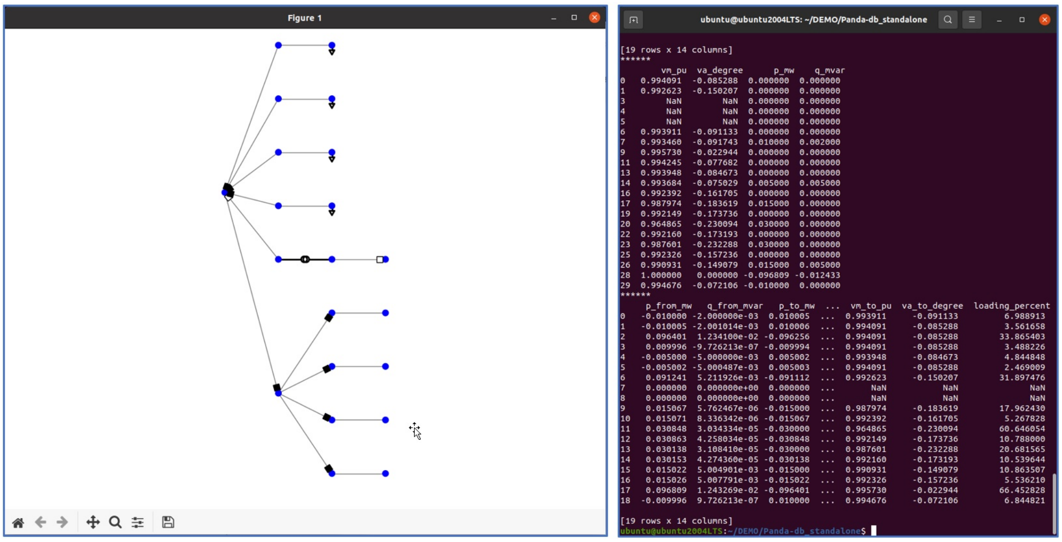

Figure 23

(2) Dynamic system

Ensure the database is set up and accessible before running this.

python3 Simulator.py

Output:

```
{'EPIC/400 V/Gen/CN': 0, 'EPIC/400 V/Trans/CN': 1, 'EPIC/400 V/T1/CBRCTRCN': 2, 'EPIC/400 V/Ext. Grid/CN':
 3, 'EPIC/400 V/T2/CBRPTRCN': 4, 'EPIC/400 V/T3/PTRCBRCN': 5, 'EPIC/400 V/G3/CONCBRCN': 6, 'EPIC/400 V/G3/
CBRCTRCN': 8, 'EPIC/400 V/G2/GENCBRCN': 9, 'EPIC/400 V/G2/CBRCTRCN': 10, 'EPIC/400 V/G1/GENCBRCN': 11, 'EP
IC/400 V/G1/CBRCTRCN': 12, 'EPIC/400 V/G4/CONCBRCN': 13, 'EPIC/400 V/G4/CBRCTRCN': 15, 'EPIC/400 V/L1/OHLC
TRCN': 16, 'EPIC/400 V/L1/CTRCBRCN': 18, 'EPIC/400 V/L2/OHLCTRCN': 19, 'EPIC/400 V/L2/CTRCBRCN': 21, 'EPIC
/400 V/L3/OHLCTRCN': 22, 'EPIC/400 V/L3/CTRCBRCN': 24, 'EPIC/400 V/L4/MOTCTRCN': 25, 'EPIC/400 V/L4/CTRCBR
CN': 27}
1
numba cannot be imported and numba functions are disabled.
Probably the execution is slow.
Please install numba to gain a massive speedup.
(or if you prefer slow execution, set the flag numba=False to avoid this warning!)

{'bus':        vm_pu  va_degree      p_mw    q_mvar
0   0.994091  -0.085288  0.000000  0.000000
1   0.992623  -0.150207  0.000000  0.000000
3        NaN        NaN  0.000000  0.000000
4        NaN        NaN  0.000000  0.000000
5        NaN        NaN  0.000000  0.000000
6   0.993911  -0.091133  0.000000  0.000000
7   0.993460  -0.091743  0.010000  0.002000
9   0.995730  -0.022944  0.000000  0.000000
11  0.994245  -0.077682  0.000000  0.000000
13  0.993948  -0.084673  0.000000  0.000000
14  0.993684  -0.075029  0.005000  0.005000
16  0.992392  -0.161705  0.000000  0.000000
17  0.987974  -0.183619  0.015000  0.000000
19  0.992149  -0.173736  0.000000  0.000000
20  0.964865  -0.230094  0.030000  0.000000
22  0.992160  -0.173193  0.000000  0.000000
23  0.987601  -0.232288  0.030000  0.000000
25  0.992326  -0.157236  0.000000  0.000000
26  0.990931  -0.149079  0.015000  0.005000
28  1.000000   0.000000 -0.096809 -0.012433
29  0.994676  -0.072106 -0.010000  0.000000, 'line':     p_from_mw   q_from_mvar   p_to_mw  ...  vm_to_pu
 va_to_degree  loading_percent
0   -0.010000 -2.000000e-03  0.010005  ...  0.993911     -0.091133        6.988913
1   -0.010005 -2.001014e-03  0.010006  ...  0.994091     -0.085288        3.561658
2    0.096401  1.234100e-02 -0.096256  ...  0.994091     -0.085288       33.865403
3    0.009996 -9.726213e-07 -0.009994  ...  0.994091     -0.085288        3.488226
4   -0.005000 -5.000000e-03  0.005002  ...  0.993948     -0.084673        4.844848
5   -0.005002 -5.000487e-03  0.005003  ...  0.994091     -0.085288        2.469009
6    0.091241  5.211926e-03 -0.091112  ...  0.992623     -0.150207       31.897476
7    0.000000  0.000000e+00  0.000000  ...       NaN           NaN             NaN
8    0.000000  0.000000e+00  0.000000  ...       NaN           NaN             NaN
9    0.015067  5.762467e-06 -0.015000  ...  0.987974     -0.183619       17.962430
10   0.015071  8.336342e-06 -0.015067  ...  0.992392     -0.161705        5.267828
11   0.030848  3.034334e-05 -0.030000  ...  0.964865     -0.230094       60.646054
12   0.030863  4.258034e-05 -0.030848  ...  0.992149     -0.173736       10.788000
13   0.030138  3.108410e-05 -0.030000  ...  0.987601     -0.232288       20.681565
14   0.030153  4.274360e-05 -0.030138  ...  0.992160     -0.173193       10.539644
15   0.015022  5.004901e-03 -0.015000  ...  0.990931     -0.149079       10.863507
16   0.015026  5.007791e-03 -0.015022  ...  0.992326     -0.157236        5.536210
17   0.096809  1.243269e-02 -0.096401  ...  0.995730     -0.022944       66.452828
18  -0.009996  9.726213e-07  0.010000  ...  0.994676     -0.072106        6.844821

[19 rows x 14 columns], 'switch':     bus  element et  type  closed  name  z_ohm
0     6        1  l  None    True  None    0.0
1     9        2  l  None    True  None    0.0
2    11        3  l  None    True  None    0.0
3    13        5  l  None    True  None    0.0
4     0        6  l  None    True  None    0.0
5     3        7  l  None   False  None    0.0
6     1        8  l  None   False  None    0.0
7     1       10  l  None    True  None    0.0
8     1       12  l  None    True  None    0.0
9     1       14  l  None    True  None    0.0
10    1       16  l  None    True  None    0.0, 'generator':    name  bus  p_mw  vm_pu  sn_mva  ...  slack
  in_service  slack_weight  type  power_station_trafo
0  None   28   0.5    1.0     NaN  ...   True        True           0.0  None                 NaN

[1 rows x 13 columns], 'load':    name  bus   p_mw  q_mvar  const_z_percent  const_i_percent  sn_mva  scal
ing  in_service type
0  None   17  0.015   0.000             0.0              0.0     NaN      1.0        True  wye
1  None   20  0.030   0.000             0.0              0.0     NaN      1.0        True  wye
2  None   23  0.030   0.000             0.0              0.0     NaN      1.0        True  wye
3  None   26  0.015   0.005             0.0              0.0     NaN      1.0        True  wye, 'trafo':
   name std_type  hv_bus  lv_bus  sn_mva  ...  parallel   df  in_service  pt_percent   oltc
0  None     None       4       5     0.1  ...         1  1.0        True         NaN  False

[1 rows x 25 columns]}
```

Figure 24

8 Larger Scale Model

The following utilities can be used to create the artifacts required for larger scale models.

8.1 SSD Merger

This utility merges multiple single-substation SSD files into one multi-substations SSD file given the SED files.
To use:

(1) Place single-substation SSD files and respective SED files in ssd_sed_input_files folder

(2) Run:
python3 merge_ssd.py

(3) The merged SSD file will be created in the ssd_output_file folder

8.2 SCD Merger

This utility merges mulitple single-substation SCD files into one multi-substations SCD file.
To use:

(1) Place single-substation SCD files in scd_input_files folder

(2) Run:
python3 merge_scd.py

(3) The merged SCD file will be created in scd_output_files folder

Edit merge_scd.py to point to input and output SCD files of your choice.

8.3 SCD Parser

This utility converts an SCD file to JSON format, which can then be used as an input file for Mininet.
To run:

- python3 parse.py

Edit parse.py to point to the SCD and JSON files of your choice.

8.4 Thresholds

The file Thresholds.xml contains the threshold values for all IEDs. It can be created manually or with the script below to help create one using a threshold template.
Input files required from user to be placed in working_folder:

- <IED_name>.icd for every IED (can be placed under sub-folders)

To run:

- python3 create_thresholds.py

The following file will be generated in working_folder:

- Thresholds.xml

9 Acknowledgement

This work is supported by the National Research Foundation, Singapore, Singapore University of Technology and Design under its National Satellite of Excellence in Design Science and Technology for Secure Critical Infrastructure Grant (NSoE_DeST-SCI2019-0005).