PNNL-31217

Universal Utility Data Exchange (UUDEX) Phase 4 Demonstration Environment and Results

Cybersecurity of Energy Delivery Systems (CEDS) Research and Development

April 2021

SR Mix
MJ Rice
JD Welsh
SE Harpool
S Niddodi
NS Van

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

PNNL-31217

Universal Utility Data Exchange (UUDEX) Phase 4 Demonstration Environment and Results

Cybersecurity of Energy Delivery Systems (CEDS) Research and Development

April 2021

SR Mix
MJ Rice
JD Welsh
SE Harpool
S Niddodi
NS Van

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

Revision History

Revision	Date	Deliverable (Reason for Change)	Release #
0	4/30/2021	Initial Release	PNNL-31217

Summary

This report summarizes the Universal Utility Data Exchange (UUDEX) Phase 4 demonstration environment and tasks performed during the Phase 4 demonstration.

Acronyms and Abbreviations

ACL	access control list
API	application programming interface
CA	certificate authority
CIM	IEC 61970 Common Information Model
CSV	comma separated values
DOE	U. S. Department of Energy
EMS	Energy Management System
HTTP	hypertext transfer protocol
IAB	Industry Advisory Board
IEC	International Electrotechnical Commission
JSON	JavaScript Object Notation
mRID	IEC 61970 CIM Master Resource Identifiers
MW	megawatt
MVAR	mega volt ampere reactive
OE-417	DOE Electric Disturbance Events Report Form
PDF	portable document format
PNNL	Pacific Northwest National Laboratory
SSL	secure sockets layer
STIX™	Structured Threat Information eXpression
TCP	transmission control protocol
TLS	transport layer security
UI	user interface
UUDEX	Universal Utility Data Exchange

Contents

Figures

1.0 Introduction

This report provides an overview of the environment used by the Pacific Northwest National Laboratory (PNNL) in the Universal Utility Data Exchange (UUDEX) Phase 4 demonstration, and description of the demonstrations performed in the environment.

The UUDX Phase 4 demonstration took place on March 31, 2021, with representatives from PNNL and the two project sub-contractors, OATI and MITRE, performing the demonstration for members of the UUDEX Industry Advisory Board (IAB). The demonstration tasks were based on the previous Phase 3 demonstration but conducted in a multi-site environment. Additional demonstrations showed features that were developed since the Phase 3 demonstration.

The development environment was constructed by PNNL programming staff using the previously developed UUDEX Functional Design, Protocol Design, and Workflow Design documents as a starting point. The environment was augmented with subject creation management and access control security data structures and workflows, and data exchange structure documents developed as part of Phase 3 and Phase 4.

2.0 Demonstration Environment

The demonstration environment consists of three UUDEX prototype servers and several UUDEX prototype publisher and subscriber clients. The UUDEX prototype servers were located at three sites across the United States and implemented using two different underlying message bus applications.

2.1 Multi-site Environment

As shown in Figure 1, the UUDEX Phase 4 demonstration environment consists of three UUDEX servers each located at different sites in the United States, all connected to the public Internet.

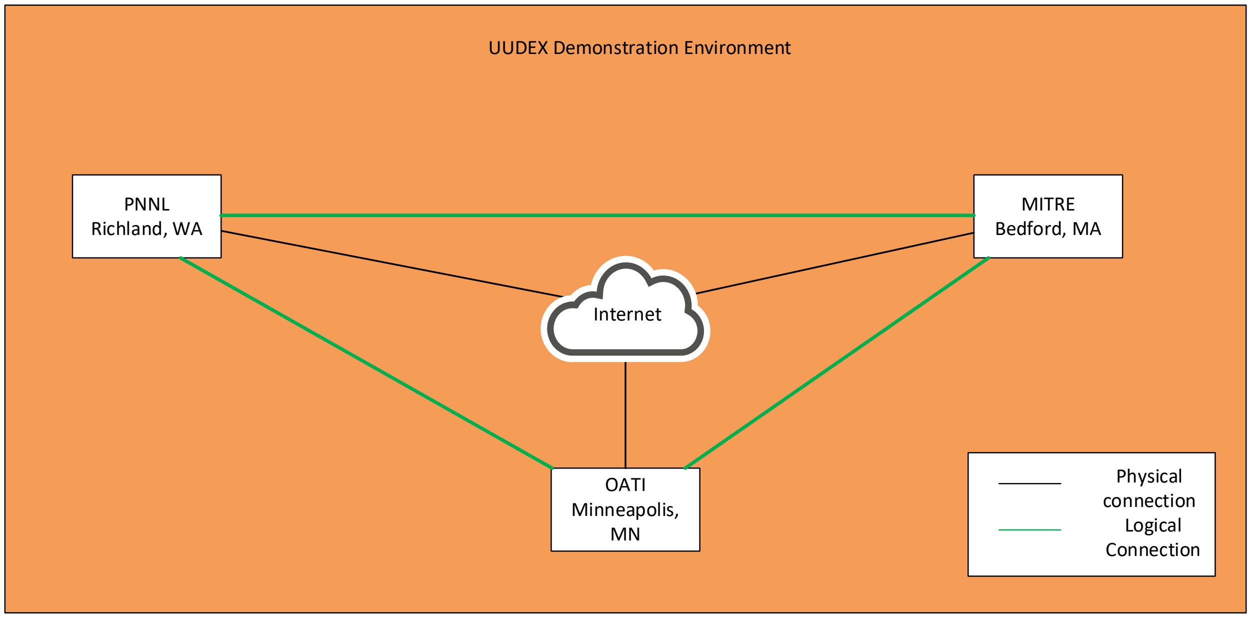


Figure 1. UUDEX Phase 4 Demonstration Environment

The UUDEX prototype server located at the PNNL Richland, WA campus uses the RabbitMQ message bus.

The UUDEX prototype server located at the OATI Minneapolis, MN facility uses the Apache Kafka message bus.

The UUDEX prototype server located at the MITRE Bedford, MA facility uses the RabbitMQ message bus.

UUDEX publish and subscribe clients were located at each of the three locations. Note that the client interface to the UUDEX server uses a message-bus independent interface, so the client does not need to be aware of the underlying message bus implementation to use the services of the UUDEX prototype server.

Firewall rulesets were configured at each location to only allow access from the other two UUDEX prototype server locations using port TCP/443 (transmission control protocol port for the transport layer security [TLS]) for information exchange.

2.2 UUDEX Prototype Server - RabbitMQ

The UUDEX prototype server at PNNL was implemented on a Linux Workstation (Red Hat Enterprise Linux Workstation release 7.9) configured in a VMware ESXi version 6.5 virtual machine with 60 GB of disk, 2GB of ram, and a single Intel processor. The UUDEX prototype server application was implemented in Python using the “gunicorn” (Green Unicorn) hypertext transfer protocol (HTTP) server v19.9. The gunicorn server is front ended with “nginx” v1.16.1 to handle TLS and process the digital certificates. The UUDEX prototype server maintained its internal configuration in a private database (i.e., a database that is not exposed to the UUDEX clients) using PostgreSQL v12.

The UUDEX prototype server at PNNL uses the RabbitMQ™ v3.8 message bus for accepting published messages and fulfilling subscriptions. RabbitMQ™ performs all the message processing, including storing published messages until the subscriber client requests them.

All UUDEX prototype server code was implemented using Python v3.6.8.

The UUDEX prototype server also served as the UUDEX Identity Authority running a certificate authority (CA) used to create and verify the X.509 digital certificates used in the demonstration. The CA was simulated and created specifically for the demo. It consisted of a CA certificate and private key. The OpenSSL tool, v1.0.2k-fips, was used to create these two items.

The development environment consisted of PyCharm running on Microsoft Windows 10 and RabbitMQ™ and PostgreSQL database running remotely on the UUDEX prototype server during development.

Flask v1.1.2 is a Web Framework that was used to develop the server. This applies to the server only and does not apply to the client.

The UUDEX prototype server at MITRE was implemented on a Linux Workstation (Red Hat Enterprise Linux Workstation release 7.9) in a VMware ESXi version 6.7 virtual machine allocated with two Intel CPU cores, 10GB of disk, and 8GB of RAM. All other software was identical to the PNNL implementation.

2.3 UUDEX Prototype Server – Apache Kafka

The UUDEX prototype server at OATI was implemented on a Linux server (CentOS Linux 7.9) with two 4-core Intel CPUs (2.5GHz), 4GB RAM, and two 73GB SAS drives in a RAID-1 mirror configuration.

The UUDEX prototype server at OATI uses the Apache Kafka v2.7 message bus for accepting published messages and fulfilling subscriptions. As with the RabbitMQ™ implementation, Apache Kafka performs all the message processing, including storing published messages until the subscriber client requests them.

OATI modified the software that processes the UUDEX client calls arriving on the UUDEX Server REST interface to be compatible with Apache Kafka, allowing the client interface to be identical, even though RabbitMQ and Kafka have slightly different internal requirements.

All other software was identical between the RabbitMQ and Kafka implementations.

2.4 UUDEX Prototype Clients

The UUDEX prototype clients at PNNL consisted of a combination of Microsoft Windows 10 workstations and Linux workstations (Red Hat Enterprise Linux Workstation release 7.9) residing on the same network as the UUDEX prototype server. The UUDEX prototype server node also served as a UUDEX prototype client for some demonstration tasks. The UUDEX prototype clients could be either publishers or subscribers, based on the particular code used for a demonstration task.

UUDEX prototype client code for both publisher and subscriber was implemented using Python v3.7.8

The UUDEX prototype client software was developed using the same development environment as was used for the server, described above.

A demonstration user interface was developed and written using the Python ncurses character-based library.

UUDEX clients at OATI and MITRE used the same codebase as PNNL but ran on the same platforms as the UUDEX prototype servers at those locations.

For demonstration tasks involving simulated exchange of power system measurement data, UUDEX clients are also able to invoke the PowerWorld Simulator 20 program with the SimAuto add-on through the command-line interface. Two separate UUDEX clients were used, representing a Transmission Operator exchanging power system measurement data with a Reliability Coordinator. Both UUDEX prototype client applications were running on the same computer host running Microsoft Windows 10, and shared a single PowerWorld Simulator license to run the necessary power flows. A Python script was developed to interface between PowerWorld SimAuto add-on and the UUDEX prototype client to publish and subscribe to data.

3.0 Demonstrations Performed

The demonstration was conducted virtually using Microsoft Teams to share the screens of the presenters. The format of the demonstration consisted of an introduction to the particular demonstration test using the PowerPoint presentation found in Appendix A, followed by individual presenters sharing their screens to execute the demonstration. In the case of the multi-site demonstrations, screens were shared from multiple sites in turn to show the demonstrated features.

The demonstrations included the following topics:

1. The multi-site capability of UUDEX
2. Exchange of power system measurement data
3. Verification of encrypted data transmission
4. Addition of a new UUDEX client participant
5. Security
6. Performance Management
7. Example Web-based User Interface

Prior to the start of the demonstrations, background information was presented to familiarize the IAB with terms and concepts. These are presented as slides 7 through 10 in Appendix A.

3.1 Multi-site Capabilities

The UUDEX multi-site capability demonstration consisted of clients at each of the three demonstration sites interfacing with a UUDEX Server located at one of the three sites.

For this demonstration, users at each of the sites were provisioned with digital certificates that allowed them to access the UUDEX prototype servers at each of the sites. The users and host locations are:

- Jeff – PNNL
- ScottH – PNNL
- Shashank – OATI
- Ted – MITRE

Each user was provisioned three digital certificates, one for each UUDEX prototype server at PNNL, OATI, and MITRE.

The digital certificates were identified using the convention “user@site”, i.e., the digital certificate “Shashak@PNNL” represents user Shashank accessing the UUDEX server at PNNL.

In the figures and descriptions, individual client applications are designated by numbers (e.g., 1, 2, 3), while the UUDEX server instance is shown as the icon 🛢. In most cases, a single client application is used at each location, but for the exchange of power system data, three separate applications are used at the PNNL location.

3.1.1 Unrestricted Exchange of OE-417 Report

The multi-site environment for unrestricted exchange of a DOE (Department of Energy) OE-417 reports is shown in Figure 2. For this demonstration, user Shashank at OATI wishes to submit and share OE-417 reports with other users using the UUDEX prototype server (instance) located at PNNL. The UUDEX instance at PNNL has been provisioned to allow Shashank to create and publish information.

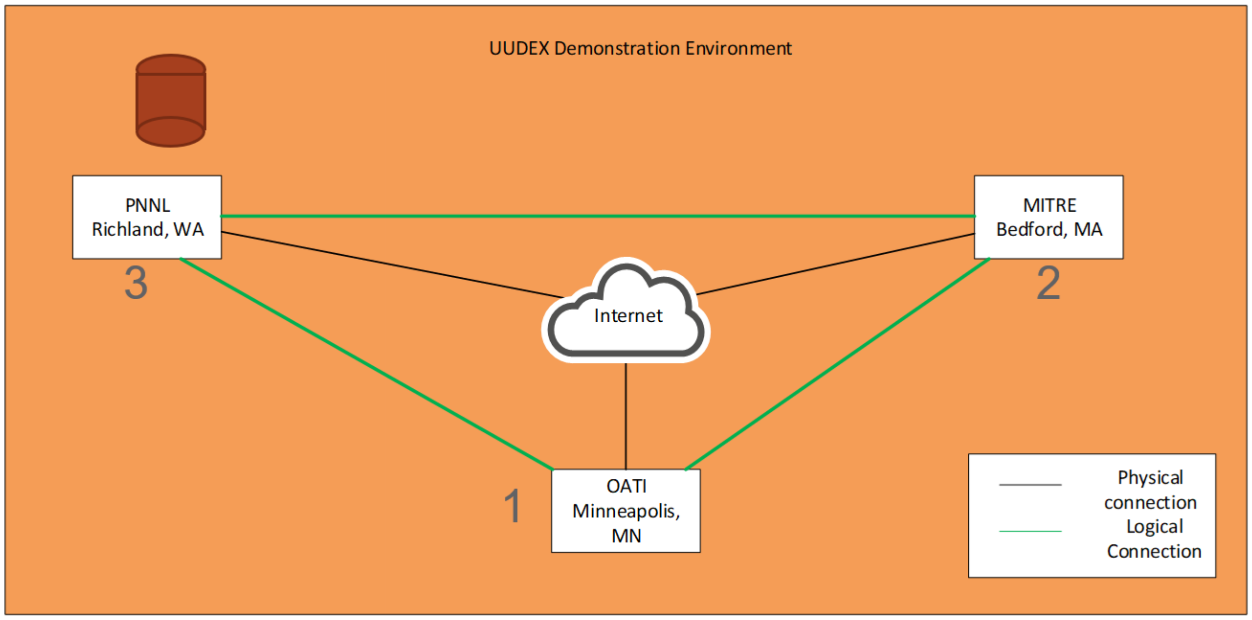


Figure 2. Unrestricted Exchange of OE-417 Report Environment

The demonstration consisted of the following steps:

- Shashank@PNNL (1) Creates a Subject (with no restrictions) at the PNNL Instance
- Shashank@PNNL (1) Discovers the Subject to verify that It has been created
- Ted@PNNL (2) Discovers the Subject to verify that he has access to it
- Jeff@PNNL (3) Discovers the Subject to verify that he has access to it
- Shashank@PNNL (1) Publishes an OE-417 PDF (portable document format) report to the Subject at the PNNL Instance
- Jeff@PNNL (3) Subscribes to the Subject, downloads the information, verifies the file is created, and displays file contents using the Adobe Acrobat reader
- Ted@PNNL (2) Subscribes to the Subject, downloads the information, and verifies the file is created (note Ted does not have access to the Adobe Acrobat reader to view the information)

This demonstration verifies that a single Publish request can be subscribed by multiple Endpoints.

3.1.2 Restricted Exchange of STIX Report

The multi-site environment for restricted exchange of STIX™ reports is shown in Figure 3. For this demonstration, user Ted at MITRE wishes to submit and share STIX reports with users at OATI but not at PNNL using the UUDEX prototype server (instance) located at OATI. The UUDEX instance at OATI has been provisioned to allow Ted to create and publish information.

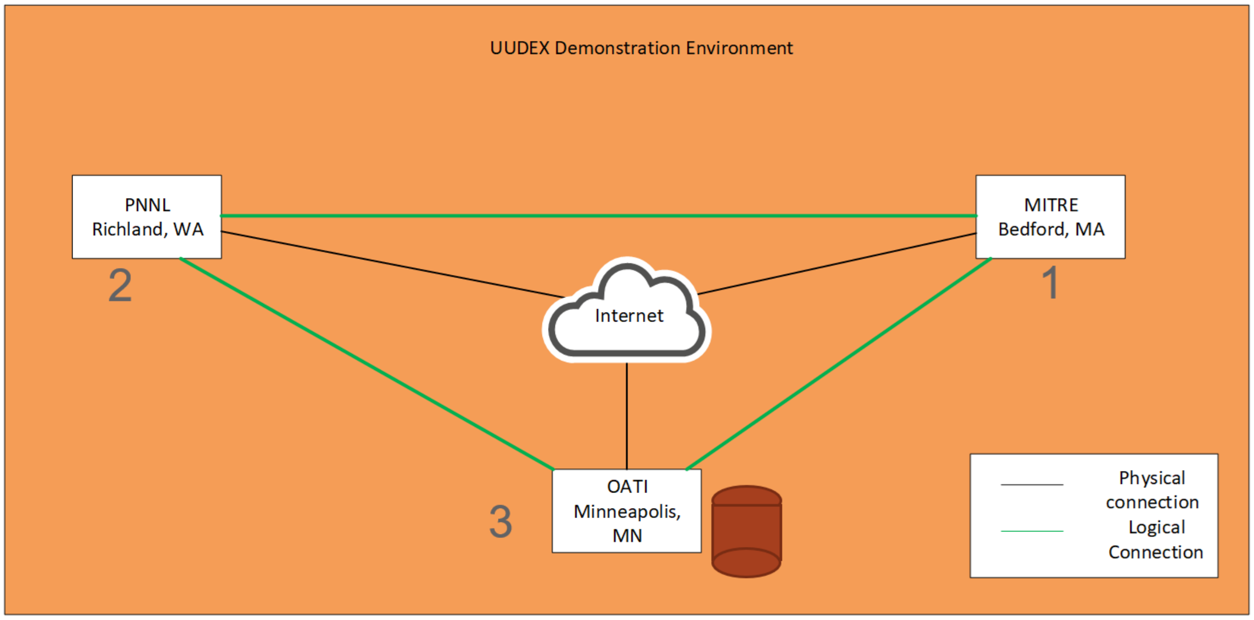


Figure 3. Restricted Exchange of STIX Information Environment

The demonstration consisted of the following steps:

- Ted@OATI (1) Creates a Subject (with restrictions allowExcept=PNNL) at OATI Instance
- Ted@OATI (1) Discovers the Subject to verify that it has been created
- Jeff@OATI (2) Attempts to Discover Subject However, since no user at PNNL has access, no information about the subject is returned indicating that Jeff does not have access
- Shashank@OATI (3) Discovers the Subject to verify that he has access
- Ted@OATI (1) Publishes a STIX file to the Subject
- Shashank@OATI (3) Subscribes to the Subject, verifies the file is created, and displays the data

This demonstration verifies that data can be published to a Common UUDEX Instance, but only specified Endpoints can Subscribe to it.

3.2 Exchange of Power System Measurement Data

The multi-site environment for exchange of power system measurement data reports is shown in Figure 4. For this demonstration, user ScottH at PNNL wishes to submit and share power system measurement data with other users using the UUDEX prototype server (instance) located at MITRE[1]. The UUDEX instance at MITRE has been provisioned to allow ScottH to create and publish information.

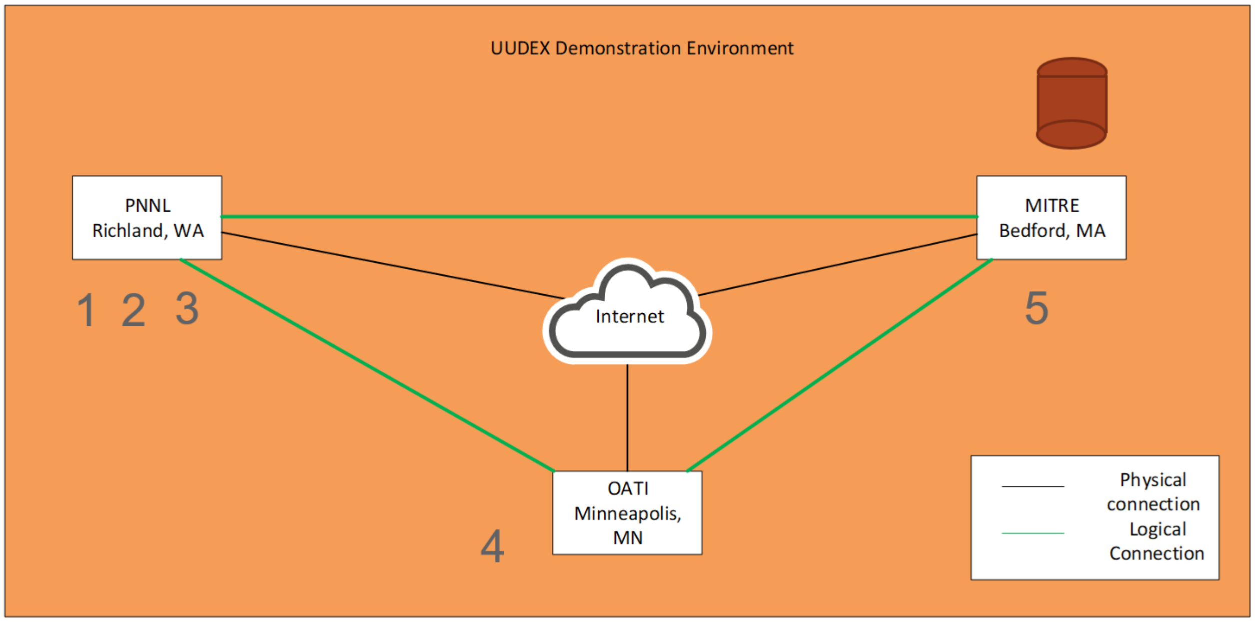


Figure 4, Exchange of Power System Measurement Data Environment

This demonstration simulates a real-world application for exchanging power system data between utilities and uses the PowerWorld PowerFlow application to simulate the actions of an Energy Management System (EMS), and an Excel spreadsheet to simulate a one-line diagram, with external data supplied via .csv (comma separated values) files. The exchanged data is formatted using JavaScript Object Notation (JSON) data structures created by UUDEX project. An example of this structure is shown in Figure 5. This structure uses the International Electrotechnical Commission (IEC) 61970 Common Information Model (CIM) Master Resource Identifiers (mRIDs) as universal identifiers, allowing that 1) Only changed data needs to be sent and 2) Data can be sent in arbitrary order. In order to simplify coding for the demonstration application, all data is transmitted every time.

In the example structure, the structure shown is for a single data value – the megawatt (MW) generation value for generator "SysGen1". Additional MW values for the same generator (e.g., ESTIMATED) could be added at the "◆" point. Additional data values, for example, the mega volt-ampere reactive (MVAR) value for the same generator could be added at the "◆◆" point, Additional generators or other power system resources could have similar blocks at the "◆◆◆" point. In the structure, not all fields are required or must be sent every time e.g., since the data is identified by the CIM mRID, the powerSystemResourceName may not be needed, or could be sent occasionally for display purposes; and the powerSystemResourceAliasName may not ever need to be sent. Additionally, the field name labels, and constants can be abbreviated to reduce communication line usage, e.g., the field name "powerSystemResourceMRID" can be abbreviated as "psrmrid", and the constant "TELEMETERED" can be abbreviated "T".

[1] Note – immediately before the demonstration, an unknown certificate error did not allow the demonstration to use the UUDEX server at MITRE, rather the UUDEX server at PNNL was used.

```
{
  "dataSet":{
    "dataElements":[
      {
        "powerSystemResource":{
          "schema":"https://www.uudex.org/uudex/0.1/powerSystemResource",
          "schemaVersion":"0.1",
          "powerSystemResourceMRID":"SysGen1",
          "powerSystemResourceName":"SystemGenerator1",
          "powerSystemResourceAliasName":"SystemGen1",
          "measurements":[
            {
              "measurementMRID":"SysGen1MW",
              "measurementName":"SystemGenerator1",
              "measurementType":"ANALOG",
              "unitSymbol":"MW",
              "values":[
                {
                  "type":"",
                  "value":71.64455056,
                  "quality":{
                    "currentSource":"TELEMETERED",
                    "validity":[
                      "VALID"
                    ]
                  },
                  "timeStamp":{
                    "quality":"VALID",
                    "value":"2021/03/24 20:02:29"
                  }
                }
              ]     ◆
            }
          ]    ◆◆
        }
      }
    ]       ◆◆◆
  }
}
```

Figure 5, Example JSON Structure for Power System Measurement Data Exchange

This demonstration also shows how subscription processing works. A subscription is comprised of one or more subjects, and the subscription is “fulfilled”, i.e., data is sent to the subscriber(s), whenever data is published to one of its Subjects. For the demonstration, PNNL both publishes and subscribes to the data using three separate “applications” running on the same computer node in separate windows. This also shows how a running application would use existing UUDEX subjects and subscriptions to process data.

The overall data flow is shown in Figure 6, while the actual application window layout is shown in Figure 7. The data starts in the upper right with the “Client 2 Publisher” application simulating field telemetry by making small changes in the value associated with a generator, and publishing the data using the UUDEX “Gen telemetered values dataset” subject. This subject is a member of the “Telemetered values” subscription.

Data is retrieved in the lower left by the “Client 1 Receiver” application that subscribes to the “Telemetered values” subscription to retrieve the simulated telemetered values published by “Client 2 Publisher”. It then solves the power flow and publishes the results to the “Gen dataset”, “Bus dataset” and “Branch dataset” subjects, which are all part of the “PowerFlow” subscription.

The “Client 2 Receiver” application subscribes to the “PowerFlow” subscription and receives data whenever “Client 2 Publisher” sends data to one of the subjects in the subscription. The

“Client 2 Receiver” then transforms the UUDEX JSON-formatted values into csv files. The data in the .csv files are periodically refreshed into the simulated one-line spreadsheet by Excel so that the one line diagram can display them. The original PowerWorld one line diagram is shown in Figure 8, and the simulated Excel one line diagram is shown in Figure 9.

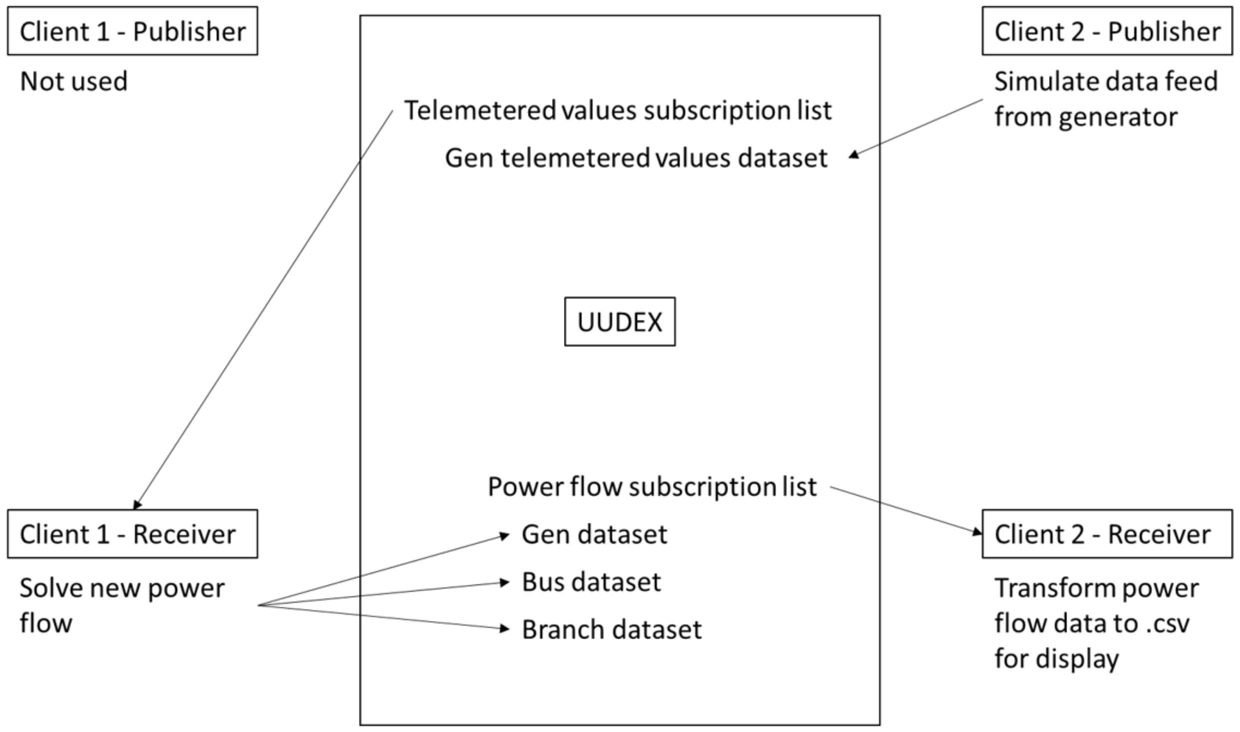
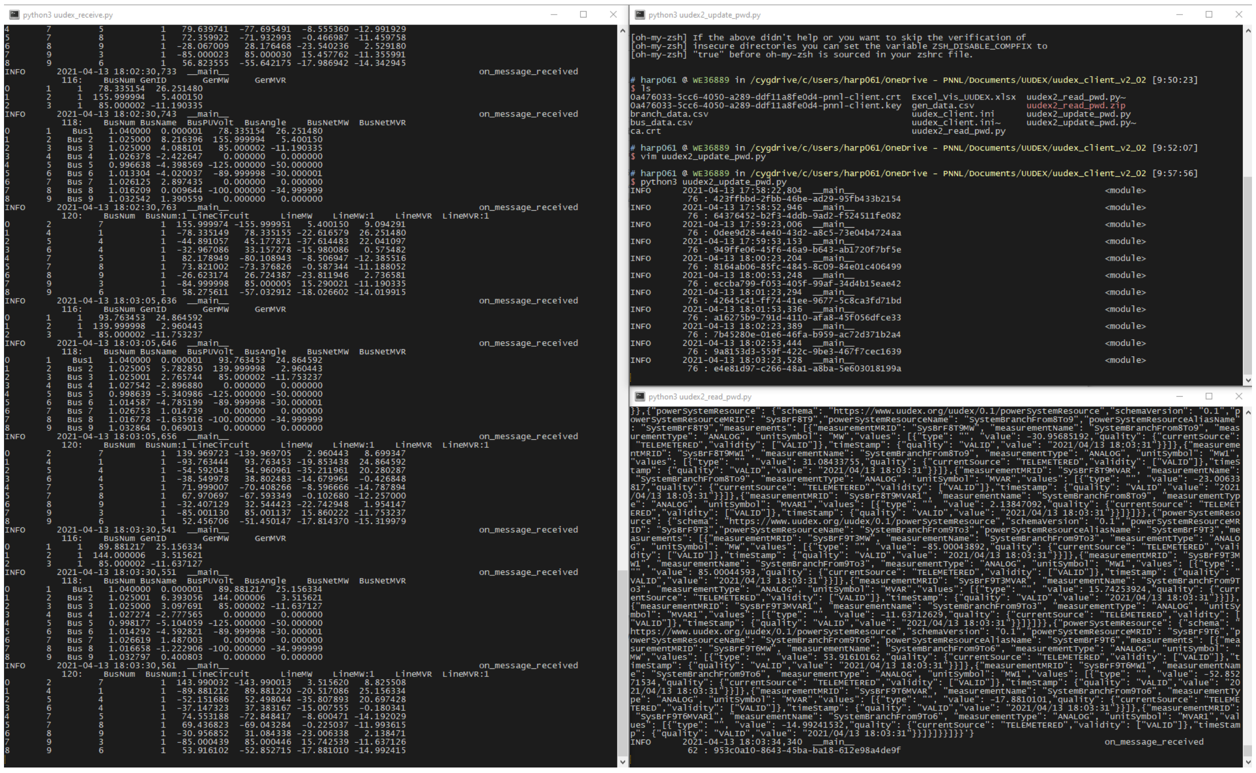

Client 1 - Publisher
Not used
Telemetered values subscription list
Gen telemetered values dataset
Client 2 - Publisher
Simulate data feed from generator
UUDEX
Power flow subscription list
Gen dataset
Bus dataset
Branch dataset
Client 1 - Receiver
Solve new power flow
Client 2 - Receiver
Transform power flow data to .csv for display

Figure 6. Application Window Layout



```
4       7         5           1   79.639741  -77.695491  -8.555360 -12.991929
5       7         8           1   72.359922  -71.932993  -0.466987 -11.459758
6       8         9           1  -28.067009   28.176468 -23.540236   2.529180
7       9         3           1  -85.000023   85.000030  15.457762 -11.355991
8       9         6           1   56.823555  -55.642175 -17.986942 -14.342945
INFO      2021-04-13 18:02:30,733  __main__                                     on_message_received
           116:    BusNum GenID      GenMW     GenMVR
0       1     1   78.335154  26.251480
1       2     1  155.999994   5.400150
2       3     1   85.000002 -11.190335
INFO      2021-04-13 18:02:30,743  __main__                                     on_message_received
           118:    BusNum BusName  BusPUVolt  BusAngle    BusNetMW  BusNetMVR
0       1    Bus1   1.040000  0.000001   78.335154  26.251480
1       2   Bus 2   1.025000  8.216396  155.999994   5.400150
2       3   Bus 3   1.025000  4.088101   85.000002 -11.190335
3       4   Bus 4   1.026378 -2.422647    0.000000   0.000000
4       5   Bus 5   0.996638 -4.398569 -125.000000 -50.000000
5       6   Bus 6   1.013304 -4.020037  -89.999998 -30.000001
6       7   Bus 7   1.026125  2.897435    0.000000   0.000000
7       8   Bus 8   1.016209  0.009644 -100.000000 -34.999999
8       9   Bus 9   1.032542  1.390559    0.000000   0.000000
INFO      2021-04-13 18:02:30,763  __main__                                     on_message_received
           120:    BusNum  BusNum:1 LineCircuit      LineMW    LineMW:1    LineMVR  LineMVR:1
0       2         7           1  155.999974 -155.999951   5.400150   9.094291
1       4         1           1  -78.335149   78.335155 -22.616579  26.251480
2       5         4           1  -44.891057   45.177871 -37.614483  22.041097
3       6         4           1  -32.967086   33.157278 -15.980086   0.575482
4       7         5           1   82.178949  -80.108943  -8.506947 -12.385516
5       7         8           1   73.821002  -73.376826  -0.587344 -11.188052
6       8         9           1  -26.623174   26.724387 -23.811946   2.736581
7       9         3           1  -84.999998   85.000005  15.290021 -11.190335
8       9         6           1   58.275611  -57.032912 -18.026602 -14.019915
INFO      2021-04-13 18:03:05,636  __main__                                     on_message_received
           116:    BusNum GenID      GenMW     GenMVR
0       1     1   93.763453  24.864592
1       2     1  139.999998   2.960443
2       3     1   85.000002 -11.753237
INFO      2021-04-13 18:03:05,646  __main__                                     on_message_received
           118:    BusNum BusName  BusPUVolt  BusAngle    BusNetMW  BusNetMVR
0       1    Bus1   1.040000  0.000001   93.763453  24.864592
1       2   Bus 2   1.025005  5.782850  139.999998   2.960443
2       3   Bus 3   1.025001  2.765744   85.000002 -11.753237
3       4   Bus 4   1.027542 -2.896880    0.000000   0.000000
4       5   Bus 5   0.998639 -5.340986 -125.000000 -50.000000
5       6   Bus 6   1.014587 -4.785199  -89.999998 -30.000001
6       7   Bus 7   1.026753  1.014739    0.000000   0.000000
7       8   Bus 8   1.016778 -1.635916 -100.000000 -34.999999
8       9   Bus 9   1.032864  0.069013    0.000000   0.000000
INFO      2021-04-13 18:03:05,656  __main__                                     on_message_received
           120:    BusNum  BusNum:1 LineCircuit      LineMW    LineMW:1    LineMVR  LineMVR:1
0       2         7           1  139.969723 -139.969705   2.960443   8.699347
1       4         1           1  -93.763444   93.763453 -19.853438  24.864592
2       5         4           1  -54.592043   54.960961 -35.211961  20.280287
3       6         4           1  -38.549978   38.802483 -14.679964  -0.426848
4       7         5           1   71.999007  -70.408266  -8.596666 -14.787894
5       7         8           1   67.970697  -67.593349  -0.102680 -12.257000
6       8         9           1  -32.407129   32.544423 -22.742948   1.954147
7       9         3           1  -85.001130   85.001137  15.860222 -11.753237
8       9         6           1   52.456706  -51.450147 -17.814370 -15.319979
INFO      2021-04-13 18:03:30,541  __main__                                     on_message_received
           116:    BusNum GenID      GenMW     GenMVR
0       1     1   89.881217  25.156334
1       2     1  144.000006   3.515621
2       3     1   85.000002 -11.637127
INFO      2021-04-13 18:03:30,551  __main__                                     on_message_received
           118:    BusNum BusName  BusPUVolt  BusAngle    BusNetMW  BusNetMVR
0       1    Bus1   1.040000  0.000001   89.881217  25.156334
1       2   Bus 2   1.025001  6.393056  144.000006   3.515621
2       3   Bus 3   1.025000  3.097691   85.000002 -11.637127
3       4   Bus 4   1.027274 -2.777565    0.000000   0.000000
4       5   Bus 5   0.998177 -5.104059 -125.000000 -50.000000
5       6   Bus 6   1.014292 -4.592821  -89.999998 -30.000001
6       7   Bus 7   1.026619  1.487003    0.000000   0.000000
7       8   Bus 8   1.016658 -1.222906 -100.000000 -34.999999
8       9   Bus 9   1.032797  0.400803    0.000000   0.000000
INFO      2021-04-13 18:03:30,561  __main__                                     on_message_received
           120:    BusNum  BusNum:1 LineCircuit      LineMW    LineMW:1    LineMVR  LineMVR:1
0       2         7           1  143.990032 -143.990013   3.515620   8.825508
1       4         1           1  -89.881212   89.881220 -20.517086  25.156334
2       5         4           1  -52.151686   52.498044 -35.807893  20.697428
3       6         4           1  -37.147323   37.383167 -15.007555  -0.180341
4       7         5           1   74.553188  -72.848417  -8.600471 -14.192029
5       7         8           1   69.436823  -69.043284  -0.225037 -11.993615
6       8         9           1  -30.956852   31.084338 -23.006338   2.138471
7       9         3           1  -85.000439   85.000446  15.742539 -11.637126
8       9         6           1   53.916102  -52.852715 -17.881010 -14.992415
```

```
[oh-my-zsh] If the above didn't help or you want to skip the verification of
[oh-my-zsh] insecure directories you can set the variable ZSH_DISABLE_COMPFIX to
[oh-my-zsh] "true" before oh-my-zsh is sourced in your zshrc file.


# harp061 @ WE36889 in /cygdrive/c/Users/harp061/OneDrive - PNNL/Documents/UUDEX/uudex_client_v2_02 [9:50:23]
$ ls
0a476033-5cc6-4050-a289-ddf11a8fe0d4-pnnl-client.crt  Excel_Vis_UUDEX.xlsx  uudex2_read_pwd.py~
0a476033-5cc6-4050-a289-ddf11a8fe0d4-pnnl-client.key  gen_data.csv          uudex2_read_pwd.zip
branch_data.csv                                       uudex_client.ini      uudex2_update_pwd.py
bus_data.csv                                          uudex_client.ini~     uudex2_update_pwd.py~
ca.crt                                                uudex2_read_pwd.py

# harp061 @ WE36889 in /cygdrive/c/Users/harp061/OneDrive - PNNL/Documents/UUDEX/uudex_client_v2_02 [9:52:07]
$ vim uudex2_update_pwd.py

# harp061 @ WE36889 in /cygdrive/c/Users/harp061/OneDrive - PNNL/Documents/UUDEX/uudex_client_v2_02 [9:57:56]
$ python3 uudex2_update_pwd.py
INFO      2021-04-13 17:58:22,804  __main__                                     <module>
           76 : 423ffbbd-2fbb-46be-ad29-95fb433b2154
INFO      2021-04-13 17:58:52,946  __main__                                     <module>
           76 : 64376452-b2f3-4ddb-9ad2-f524511fe082
INFO      2021-04-13 17:59:23,006  __main__                                     <module>
           76 : 0dee9d28-4e40-43d2-a8c5-73e04b4724aa
INFO      2021-04-13 17:59:53,153  __main__                                     <module>
           76 : 949ffe06-45f6-46a9-b643-ab1720f7bf5e
INFO      2021-04-13 18:00:23,204  __main__                                     <module>
           76 : 8164ab06-85fc-4845-8c09-84e01c406499
INFO      2021-04-13 18:00:53,248  __main__                                     <module>
           76 : eccba799-f053-405f-99af-34d4b15eae42
INFO      2021-04-13 18:01:23,294  __main__                                     <module>
           76 : 42645c41-ff74-41ee-9677-5c8ca3fd71bd
INFO      2021-04-13 18:01:53,336  __main__                                     <module>
           76 : a16275b9-791d-4110-afa8-45f056dfce33
INFO      2021-04-13 18:02:23,389  __main__                                     <module>
           76 : 7b45280e-01e6-46fa-b959-ac72d371b2a4
INFO      2021-04-13 18:02:53,444  __main__                                     <module>
           76 : 9a8153d3-559f-422c-9be3-467f7cec1639
INFO      2021-04-13 18:03:23,528  __main__                                     <module>
           76 : e4e81d97-c266-48a1-a8ba-5e603018199a
```

```
}},{"powerSystemResource": {"schema": "https://www.uudex.org/uudex/0.1/powerSystemResource","schemaVersion": "0.1","p
owerSystemResourceMRID": "SysBrF8T9","powerSystemResourceName": "SystemBranchFrom8To9","powerSystemResourceAliasName"
: "SystemBrF8T9","measurements": [{"measurementMRID": "SysBrF8T9MW", "measurementName": "SystemBranchFrom8To9", "meas
urementType": "ANALOG", "unitSymbol": "MW","values": [{"type": "", "value": -30.95685192,"quality": {"currentSource":
 "TELEMETERED","validity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]},{"measureme
ntMRID": "SysBrF8T9MW1", "measurementName": "SystemBranchFrom8To9", "measurementType": "ANALOG", "unitSymbol": "MW1",
"values": [{"type": "", "value": 31.08433755,"quality": {"currentSource": "TELEMETERED","validity": ["VALID"]},"timeS
tamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]},{"measurementMRID": "SysBrF8T9MVAR", "measurementName":
 "SystemBranchFrom8To9", "measurementType": "ANALOG", "unitSymbol": "MVAR","values": [{"type": "", "value": -23.00633
817,"quality": {"currentSource": "TELEMETERED","validity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "2021
/04/13 18:03:31"}}]},{"measurementMRID": "SysBrF8T9MVAR1", "measurementName": "SystemBranchFrom8To9", "measurementTyp
e": "ANALOG", "unitSymbol": "MVAR1","values": [{"type": "", "value": 2.13847092,"quality": {"currentSource": "TELEMET
ERED","validity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]}]}},{"powerSystemReso
urce": {"schema": "https://www.uudex.org/uudex/0.1/powerSystemResource","schemaVersion": "0.1","powerSystemResourceMR
ID": "SysBrF9T3","powerSystemResourceName": "SystemBranchFrom9To3","powerSystemResourceAliasName": "SystemBrF9T3","me
asurements": [{"measurementMRID": "SysBrF9T3MW", "measurementName": "SystemBranchFrom9To3", "measurementType": "ANALO
G", "unitSymbol": "MW","values": [{"type": "", "value": -85.00043892,"quality": {"currentSource": "TELEMETERED","vali
dity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]},{"measurementMRID": "SysBrF9T3M
W1", "measurementName": "SystemBranchFrom9To3", "measurementType": "ANALOG", "unitSymbol": "MW1","values": [{"type":
"", "value": 85.00044593,"quality": {"currentSource": "TELEMETERED","validity": ["VALID"]},"timeStamp": {"quality": "
VALID","value": "2021/04/13 18:03:31"}}]},{"measurementMRID": "SysBrF9T3MVAR", "measurementName": "SystemBranchFrom9T
o3", "measurementType": "ANALOG", "unitSymbol": "MVAR","values": [{"type": "", "value": 15.74253924,"quality": {"curr
entSource": "TELEMETERED","validity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]},
{"measurementMRID": "SysBrF9T3MVAR1", "measurementName": "SystemBranchFrom9To3", "measurementType": "ANALOG", "unitSy
mbol": "MVAR1","values": [{"type": "", "value": -11.63712629,"quality": {"currentSource": "TELEMETERED","validity": [
"VALID"]},"timeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]}]}},{"powerSystemResource": {"schema": "
https://www.uudex.org/uudex/0.1/powerSystemResource","schemaVersion": "0.1","powerSystemResourceMRID": "SysBrF9T6","p
owerSystemResourceName": "SystemBranchFrom9To6","powerSystemResourceAliasName": "SystemBrF9T6","measurements": [{"mea
surementMRID": "SysBrF9T6MW", "measurementName": "SystemBranchFrom9To6", "measurementType": "ANALOG", "unitSymbol": "
MW","values": [{"type": "", "value": 53.91610162,"quality": {"currentSource": "TELEMETERED","validity": ["VALID"]},"t
imeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]},{"measurementMRID": "SysBrF9T6MW1", "measurementNam
e": "SystemBranchFrom9To6", "measurementType": "ANALOG", "unitSymbol": "MW1","values": [{"type": "", "value": -52.852
71534,"quality": {"currentSource": "TELEMETERED","validity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "20
21/04/13 18:03:31"}}]},{"measurementMRID": "SysBrF9T6MVAR", "measurementName": "SystemBranchFrom9To6", "measurementTy
pe": "ANALOG", "unitSymbol": "MVAR","values": [{"type": "", "value": -17.8810101,"quality": {"currentSource": "TELEME
TERED","validity": ["VALID"]},"timeStamp": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]},{"measurementMRID":
 "SysBrF9T6MVAR1", "measurementName": "SystemBranchFrom9To6", "measurementType": "ANALOG", "unitSymbol": "MVAR1","val
ues": [{"type": "", "value": -14.99241532,"quality": {"currentSource": "TELEMETERED","validity": ["VALID"]},"timeStam
p": {"quality": "VALID","value": "2021/04/13 18:03:31"}}]}]}}]}}'}
INFO      2021-04-13 18:03:34,340  __main__                                     on_message_received
           62 : 953c0a10-8643-45ba-ba18-612e98a4de9f
```

Figure 7. Application Windows

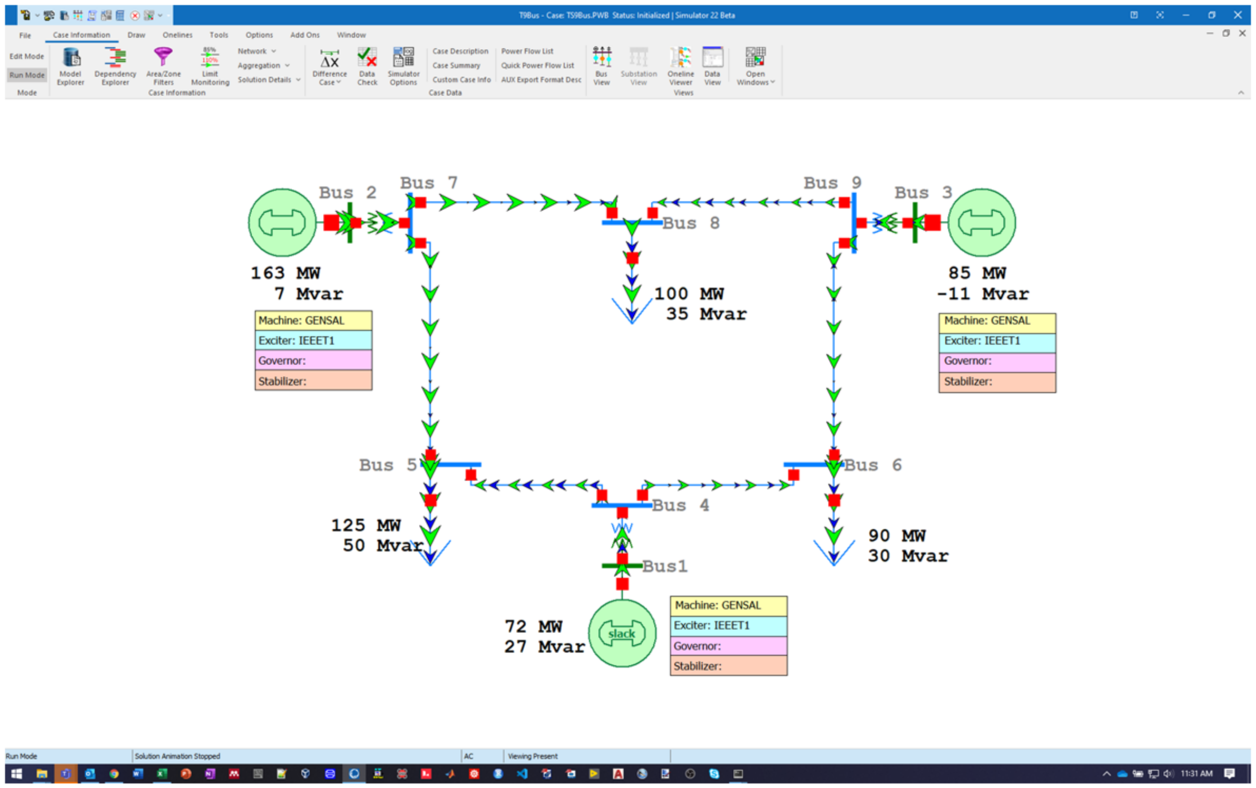


Figure 8. PowerWorld One-line

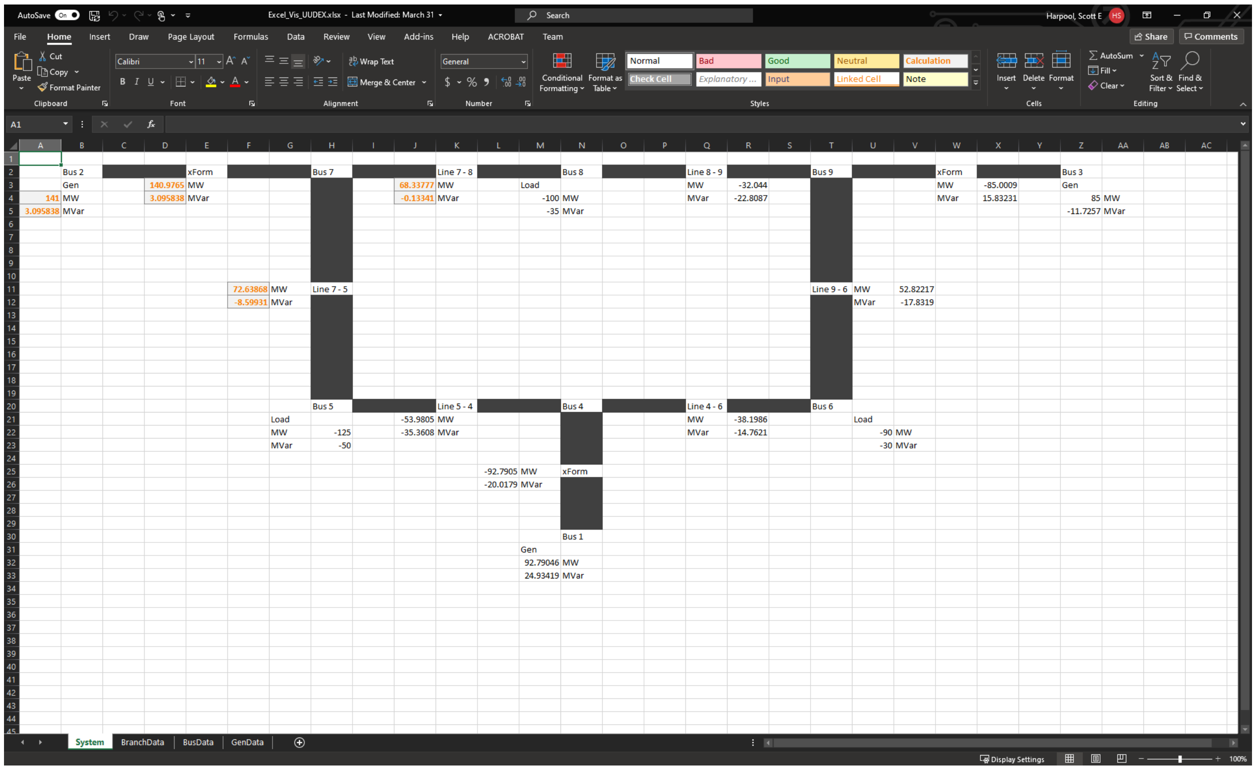


Figure 9. Excel One-line

The demonstration consisted of the following steps:

- ScottH@MITRE (1, 2, 3) uses four pre-existing Subjects at the MITRE Instance (bus, branch, generator, telemetry)
- ScottH@MITRE (1, 2, 3) uses two pre-existing Subscriptions at the MITRE Instance: one containing the telemetry Subject, and one called PowerFlow containing the "bus, branch, and generator" Subjects
- Shashank@MITRE (4) Subscribes to the PowerFlow Subscription
- Ted@MITRE (5) Subscribes to the PowerFlow Subscription
- ScottH@MITRE application (1) Subscribes to the PowerFlow Subscription
- ScottH@MITRE application (2) Publishes data to the telemetry Subject (from an application)
- ScottH@MITRE application (3) Subscribes to the telemetry Subject, executes PowerWorld, and Publishes data from PowerWorld to the bus, branch, and generator Subjects
- Shashank@MITRE (4) retrieves to the PowerFlow Subscription and views raw JSON-formatted data
- Ted@MITRE (5) retrieves to the PowerFlow Subscription and views raw JSON-formatted data
- ScottH@MITRE (1) retrieves data from the PowerFlow Subscription, processes JSON-formatted data into a csv file to view data in simulated one-line (Excel)
- All applications loop to simulate periodic telemetry, power flow execution, and data transfer

This demonstration shows how pre-existing subjects and subscriptions can be used to exchange data.

3.3 Verification of Encrypted Data Transmission

To demonstrate that the information exchanged by UUDEX is secured in transit, user Jeff at PNNL connects to the UUDEX server located at either MITRE or OATI and monitors the network traffic using the Wireshark ethernet analysis tool.

The Wireshark tool can show all the network traffic between two computer nodes. The initial exchange between the client run by user Jeff and the UUDEX prototype server is secured using TLS. The digital certificates are used to validate identity of both client and server, while encryption keys contained in digital certificates enable public key infrastructure-based encryption of all data transfers. The TLS setup negotiations are observable in the data shown in Wireshark (e.g., exchange of public keys, agreement on encryption cyphers to be used), but once encryption is enabled, all transfers are encrypted, and only encrypted data is observable. The TLS setup negotiations are only performed at the start of a set of transfers – once the authenticated and encrypted communications channel is established it is re-used making subsequent information exchanges faster (i.e., using the HTTP persistent connection standard).

Since all the exchanges are encrypted, no information about the exchanges is visible to an observer, meaning that no metadata (e.g., Header, Subject, Subscription, or Publisher data) is observable, and the only interactions that are observable are exchanges between a client and the UUDEX server; the ultimate source of subscribed data or destination of published data is not observable.

3.4 Addition of a new UUDEX client participant

The environment after the addition of a new UUDEX participant is shown in Figure 10. In this demonstration, a new UUDEX client participant (ACME-ORG) is configured and added as a client to the UUDEX prototype server at PNNL.

Note that the new participant is co-located at PNNL to minimize any firewall rule changes necessary to allow it to communicate with other UUDEX instances at OATI or MITRE.

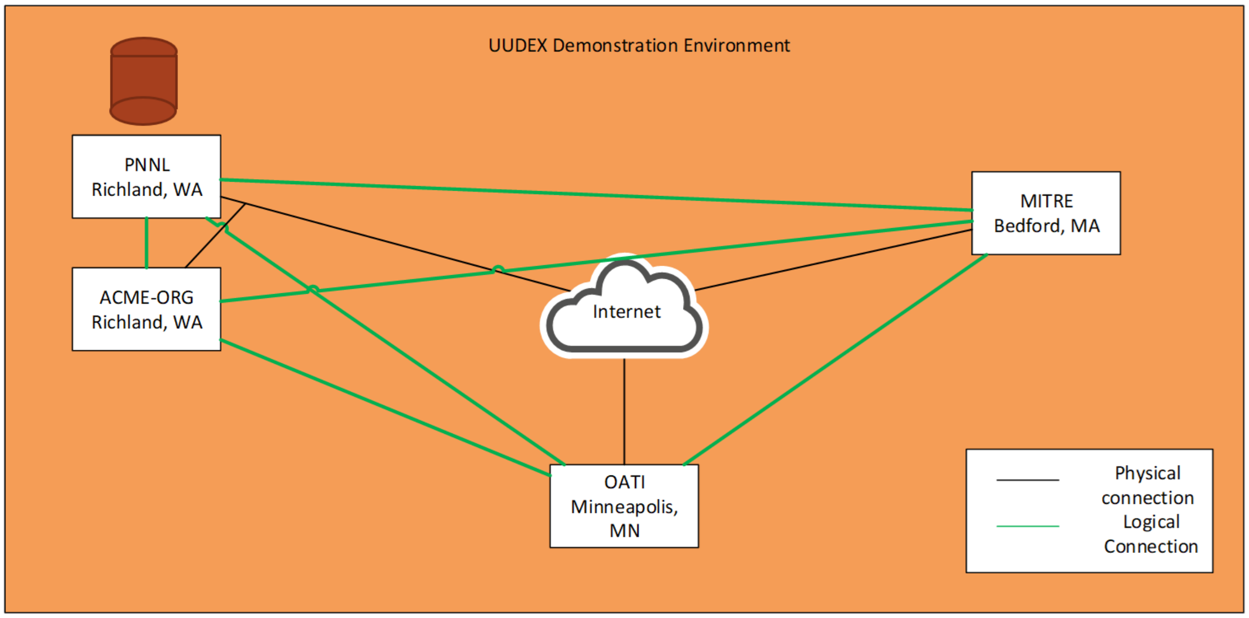


Figure 10. New UUDEX Participant Environment

The demonstration consisted of the following steps:

- Establish new Participant named “ACME-Org” for the PNNL Instance
- Provision a new ACME-Publisher Endpoint for the “ACME-Org” Participant (using PNNL CA key):
  - Generate Endpoint certificate “ACME-Publisher”
  - Install/enable certificate at PNNL – Call Create Endpoint API (application programming interface)
- Once created, ACME-ORG has established logical links to other UUDEX Participants through the UUDEX instance at PNNL: as long as the subject permissions allow it, any data published to a subject by ACME-ORG is available for subscription by MITRE or OATI, and any data published to a subject by MITRE or OATI can be subscribed to by ACME-ORG.
- As ACME-Publisher Endpoint, create a test Subject at the PNNL Instance with allowAll for Subscribe and allowAll for Discover
- As the standard PNNL Endpoint, call Subject Discovery API, find new test Subject and create a Subscription to it
- Publish to the test Subject from the new ACME-Publisher Endpoint
- Discover and subscribe to the test Subscription as the standard PNNL Endpoint
- View messages at PNNL
- Discover subject at OATI and MITRE

Once established, any subjects created by ACME-ORG on the PNNL UUDEX instance are immediately available to OATI and MITRE, and any subjects at the PNNL UUDEX server that are created without restriction are immediately available to ACME-ORG without any additional configuration changes.

3.5 Security

The security demonstration provided a high-level review of how two of the significant security features have been implemented in the UUDEX prototype.

3.5.1 Calculating the Effective ACL

This demonstration shows how permissions are applied and controlled when creating Subjects. It generally follows the steps outlined in the UUDEX Phase 4 Test Plan[1] section 3.1.11.

The function of the demonstration is to show how permissions are applied and controlled when creating Subjects, by requesting a subject be created with different specific requests and a fixed Subject Policy. The resulting “effective” access control list (ACL) is shown that represents the most restrictive “union” of the requested permissions and the permissions allowed by the Subject policy.

The demonstration shows the processing for the “Broadest Allowed Publish Access” permission but notes that processing for other permissions produce the same result.

3.5.2 Effects of ACLs on Publishing

This demonstration shows how permissions are applied when publishing to Subjects. It generally follows the steps outlined in the UUDEX Phase 4 Test Plan Section 4.4.

The demonstration shows how both implicit and explicit permissions can either allow or deny a specific publish request. It also shows that the Subject Owner has implicit access to publish to the subject even if not specified in any ACLs.

The demonstration shows the processing for the Publish access but notes that Subscribe, Discover, and Manage processing give the same result.

3.6 Performance Management

This demonstration shows how performance management features are applied when publishing to Subjects. It generally follows the steps outlined in the UUDEX Phase 4 Test Plan Section 3.1.4 and Section 4.3.1.

These demonstrations show how subjects may be created with specific performance constraints can be specified to hold either a “maximum number of messages” waiting in the queue, or a “maximum total size of all messages” waiting in queue. These constraints are used to limit the UUDEX server resources (e.g., storage space) that an individual subject queue uses.

When the performance limits are met, old messages can be dropped from the queue to make room for new messages, or the new message can be blocked from processing in the queue. Specifying the maximum number of messages and dropping old messages could be used to keep the most current copy of power system measurement data in the UUDEX datastore for later retrieval, e.g., at backup control center. Refusing new messages could be used to ensure

[1] Mix, SR, MJ Rice, S Sridhar, JD Welsh, SE Harpool, S Raju, S Yallamraju, CM Schmidt. 2020, *Universal Utility Data Exchange (UUDEX) Phase 4 Test Plan*. Pacific Northwest National Laboratory, Richland, WA.

that no messages are lost, e.g., when submitting critical reports, but this requires additional logic at the publisher to retry failed publish requests.

3.7 Example Web-based User Interface

This demonstration shows a browser application that can be used to view permissible configurations and data. The Web User Interface (UI) uses standard UUDEX Client calls to display information. A digital certificate is presented by the UI each time. The certificate is used by the UUDEX Server code to limit the returned information to only what the certificate owner has permission to view. All data presented is extracted by the Web UI from the internal data stores of the UUDEX Instance, as resident on the UUDEX Server.

The demonstration shows how a privileged user (e.g., an Administrator) has a broad view of all the data in UUDEX, while a non-privileged user can see a subset of the data.

The UI is able to show Subjects, ACLs, Participants, Endpoints, Subscriptions, and Data Sets (stored data) for any persistent messages.

The project considers this to be an administrative or diagnostic tool to view configurations and data stored in the UUDEX datastore and expects that client applications will be written to create, modify, and consume the data in a real system.

The following figures show screenshots from the Web UI:

Figure 11 shows the process for selecting the digital certificate that will be used in the Web UI user session. In this example, as shown in Figure 12, the selected certificate is for a non-privileged user named “alice”.

Figure 13 show’s Alice’s view of the list of UUDEX Participants in the UUDEX instance. Figure 14 shows a dialog box that allows the user to search for information about specific UUDEX Participants.

Figure 15 shows the list of UUDEX Endpoints associated with the selected UUDEX Participant. Recall that UUDEX Participants are organizations, while UUDEX Endpoints represent individual users, computer nodes, or applications.

Figure 16 shows the UUDEX Subject Policy associated with the selected UUDEX Participant.

Figure 17 shows the list of subjects that user Alice has permission to discover.

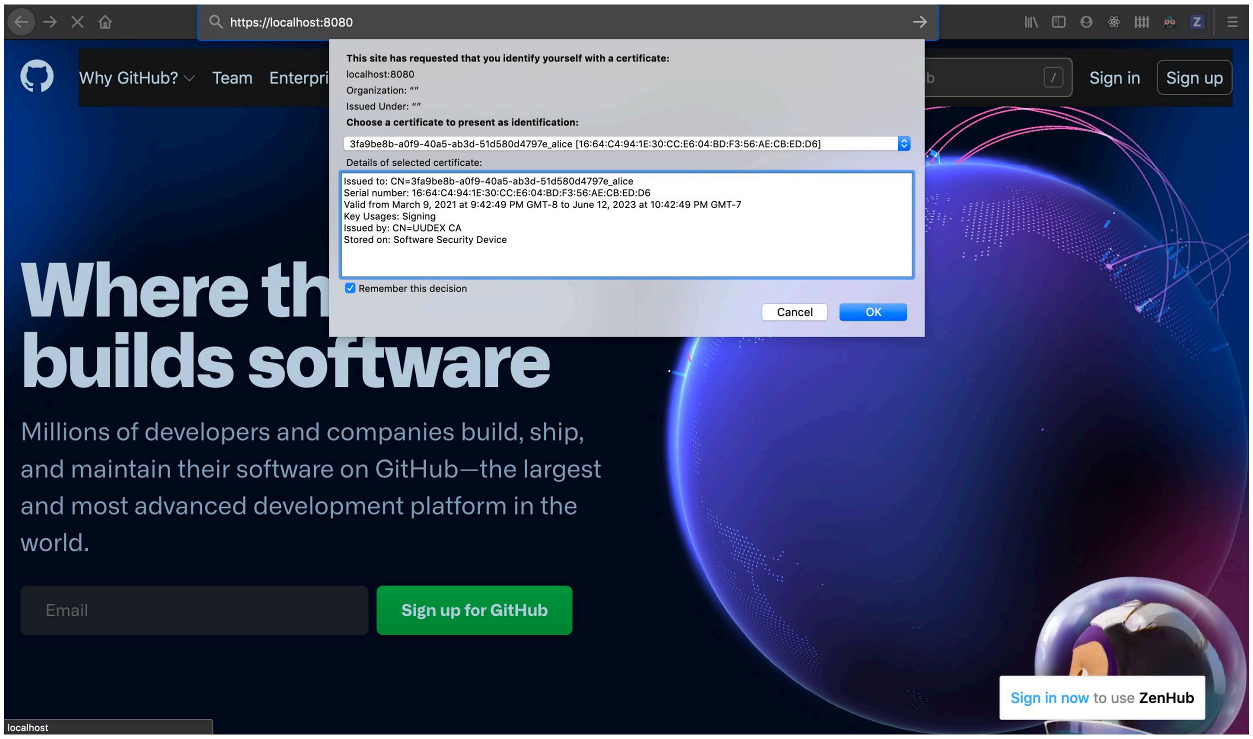
Figure 18 shows the list of active (stored) data sets for a selected subject. Once data sets are delivered through subscription fulfillment, or intentionally deleted, they will not show up in this list.

Figure 19 shows the contents (payload) of the selected data set. The payload shown is an example STIX threat-actor message.

Figure 20 shows the subscription(s) that are associated with a subject.

Figure 21 shows the effective ACL for the selected subject.

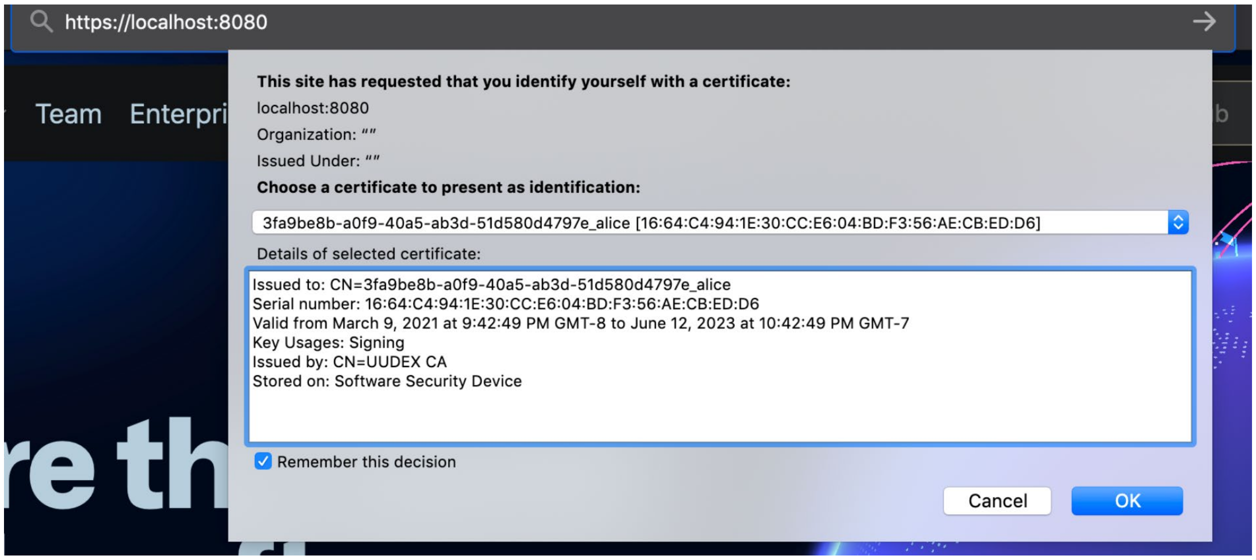
Figure 22 shows the UUDEX Subject view for an Administrative user. Note that the list of viewable subjects is larger than the same view for user Alice as shown in Figure 17.


https://localhost:8080
Why GitHub?
Team
Enterpri
Sign in
Sign up
Where th
builds software
Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.
Email
Sign up for GitHub
Sign in now to use ZenHub
localhost
Remember this decision
Cancel
OK

This site has requested that you identify yourself with a certificate:
localhost:8080
Organization: ""
Issued Under: ""
Choose a certificate to present as identification:
3fa9be8b-a0f9-40a5-ab3d-51d580d4797e_alice [16:64:C4:94:1E:30:CC:E6:04:BD:F3:56:AE:CB:ED:D6]
Details of selected certificate:

```
Issued to: CN=3fa9be8b-a0f9-40a5-ab3d-51d580d4797e_alice
Serial number: 16:64:C4:94:1E:30:CC:E6:04:BD:F3:56:AE:CB:ED:D6
Valid from March 9, 2021 at 9:42:49 PM GMT-8 to June 12, 2023 at 10:42:49 PM GMT-7
Key Usages: Signing
Issued by: CN=UUDEX CA
Stored on: Software Security Device
```

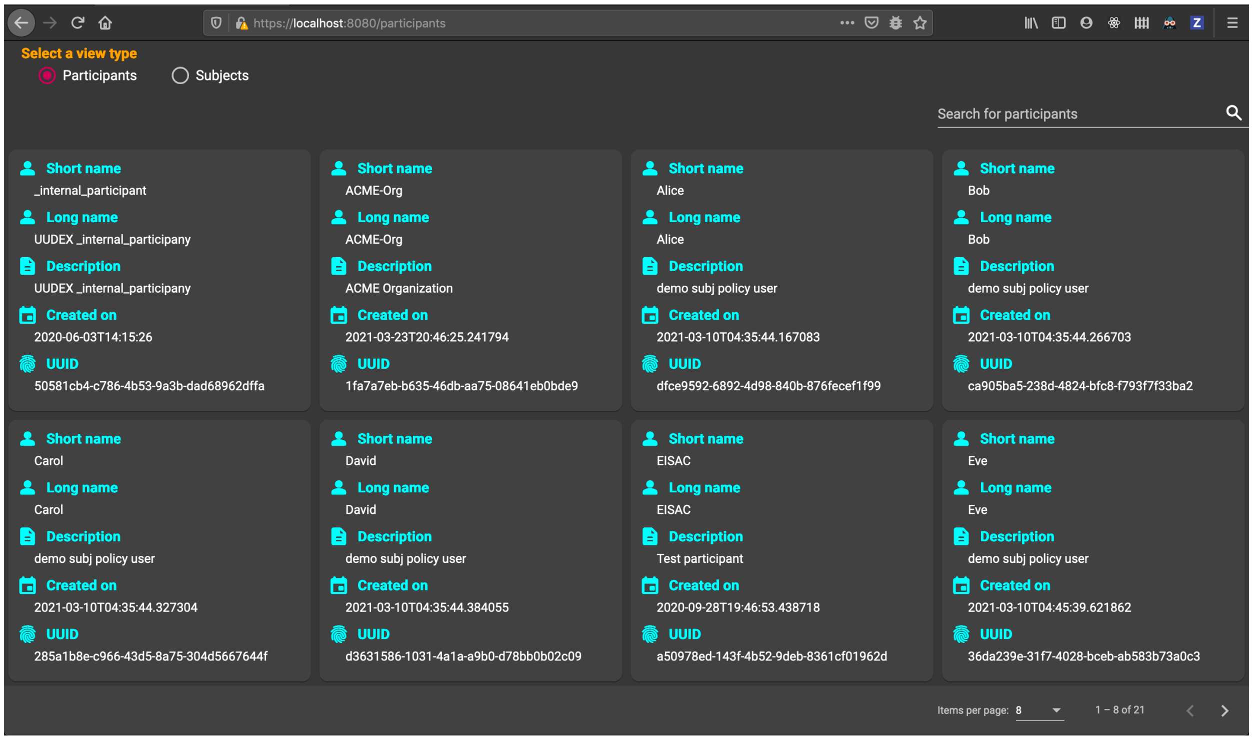
Figure 11. Web UI Certificate Selection



This site has requested that you identify yourself with a certificate:
localhost:8080
Organization: ""
Issued Under: ""
Choose a certificate to present as identification:
3fa9be8b-a0f9-40a5-ab3d-51d580d4797e_alice [16:64:C4:94:1E:30:CC:E6:04:BD:F3:56:AE:CB:ED:D6]
Details of selected certificate:

```
Issued to: CN=3fa9be8b-a0f9-40a5-ab3d-51d580d4797e_alice
Serial number: 16:64:C4:94:1E:30:CC:E6:04:BD:F3:56:AE:CB:ED:D6
Valid from March 9, 2021 at 9:42:49 PM GMT-8 to June 12, 2023 at 10:42:49 PM GMT-7
Key Usages: Signing
Issued by: CN=UUDEX CA
Stored on: Software Security Device
```

Figure 12. Web UI Certificate Selection Dialog Box



https://localhost:8080/participants

Select a view type

[✓ Participants] Subjects

Search for participants

Short name
_internal_participant
Long name
UUDEX _internal_participany
Description
UUDEX _internal_participany
Created on
2020-06-03T14:15:26
UUID
50581cb4-c786-4b53-9a3b-dad68962dffa

Short name
ACME-Org
Long name
ACME-Org
Description
ACME Organization
Created on
2021-03-23T20:46:25.241794
UUID
1fa7a7eb-b635-46db-aa75-08641eb0bde9

Short name
Alice
Long name
Alice
Description
demo subj policy user
Created on
2021-03-10T04:35:44.167083
UUID
dfce9592-6892-4d98-840b-876fecef1f99

Short name
Bob
Long name
Bob
Description
demo subj policy user
Created on
2021-03-10T04:35:44.266703
UUID
ca905ba5-238d-4824-bfc8-f793f7f33ba2

Short name
Carol
Long name
Carol
Description
demo subj policy user
Created on
2021-03-10T04:35:44.327304
UUID
285a1b8e-c966-43d5-8a75-304d5667644f

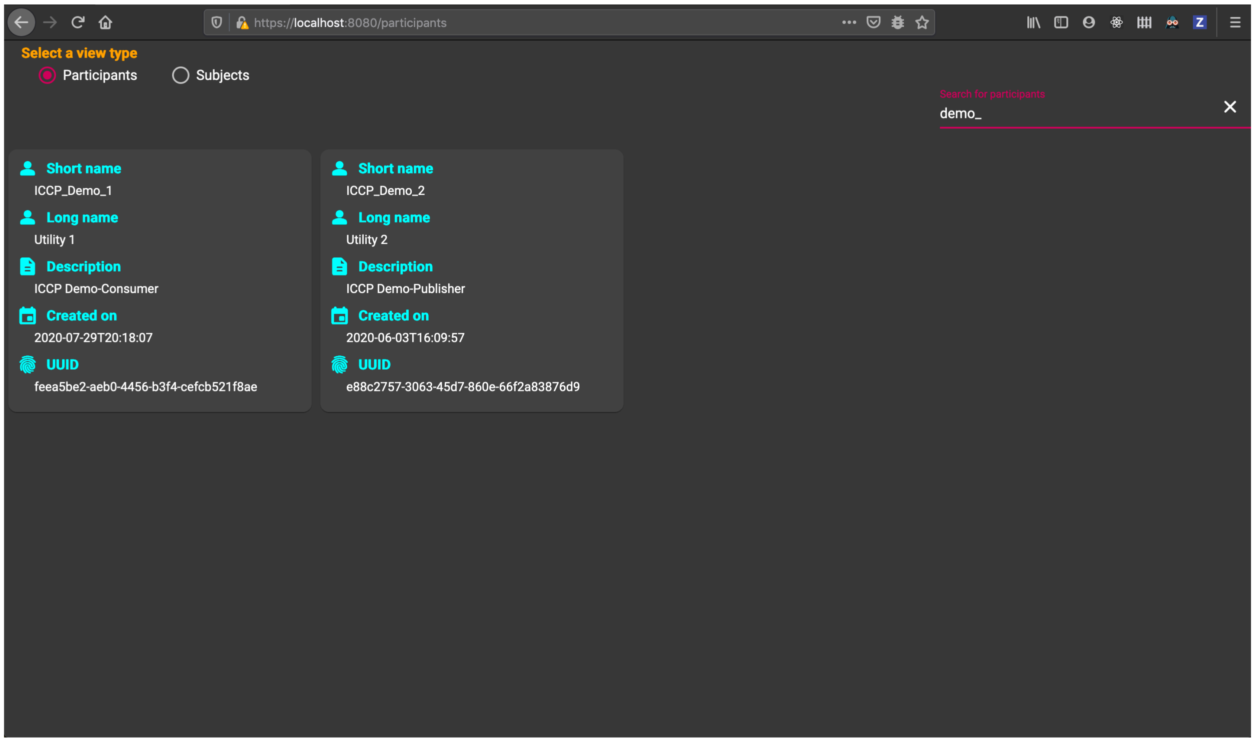
Short name
David
Long name
David
Description
demo subj policy user
Created on
2021-03-10T04:35:44.384055
UUID
d3631586-1031-4a1a-a9b0-d78bb0b02c09

Short name
EISAC
Long name
EISAC
Description
Test participant
Created on
2020-09-28T19:46:53.438718
UUID
a50978ed-143f-4b52-9deb-8361cf01962d

Short name
Eve
Long name
Eve
Description
demo subj policy user
Created on
2021-03-10T04:45:39.621862
UUID
36da239e-31f7-4028-bceb-ab583b73a0c3

Items per page: 8 1 – 8 of 21

Figure 13. Web UI Participant List



https://localhost:8080/participants

Select a view type

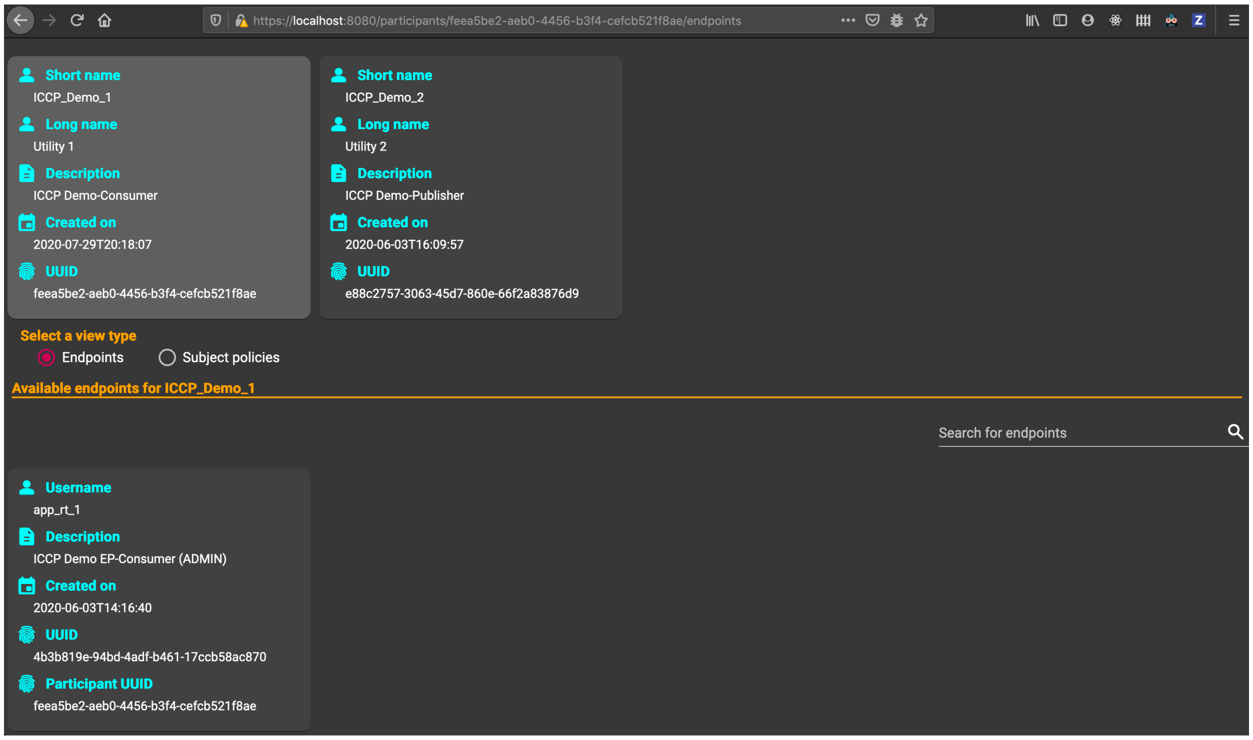
[✓ Participants] Subjects

Search for participants
demo_

Short name
ICCP_Demo_1
Long name
Utility 1
Description
ICCP Demo-Consumer
Created on
2020-07-29T20:18:07
UUID
feea5be2-aeb0-4456-b3f4-cefcb521f8ae

Short name
ICCP_Demo_2
Long name
Utility 2
Description
ICCP Demo-Publisher
Created on
2020-06-03T16:09:57
UUID
e88c2757-3063-45d7-860e-66f2a83876d9

Figure 14. Web UI Searching for Participants



https://localhost:8080/participants/feea5be2-aeb0-4456-b3f4-cefcb521f8ae/endpoints

Short name
ICCP_Demo_1
Long name
Utility 1
Description
ICCP Demo-Consumer
Created on
2020-07-29T20:18:07
UUID
feea5be2-aeb0-4456-b3f4-cefcb521f8ae

Short name
ICCP_Demo_2
Long name
Utility 2
Description
ICCP Demo-Publisher
Created on
2020-06-03T16:09:57
UUID
e88c2757-3063-45d7-860e-66f2a83876d9

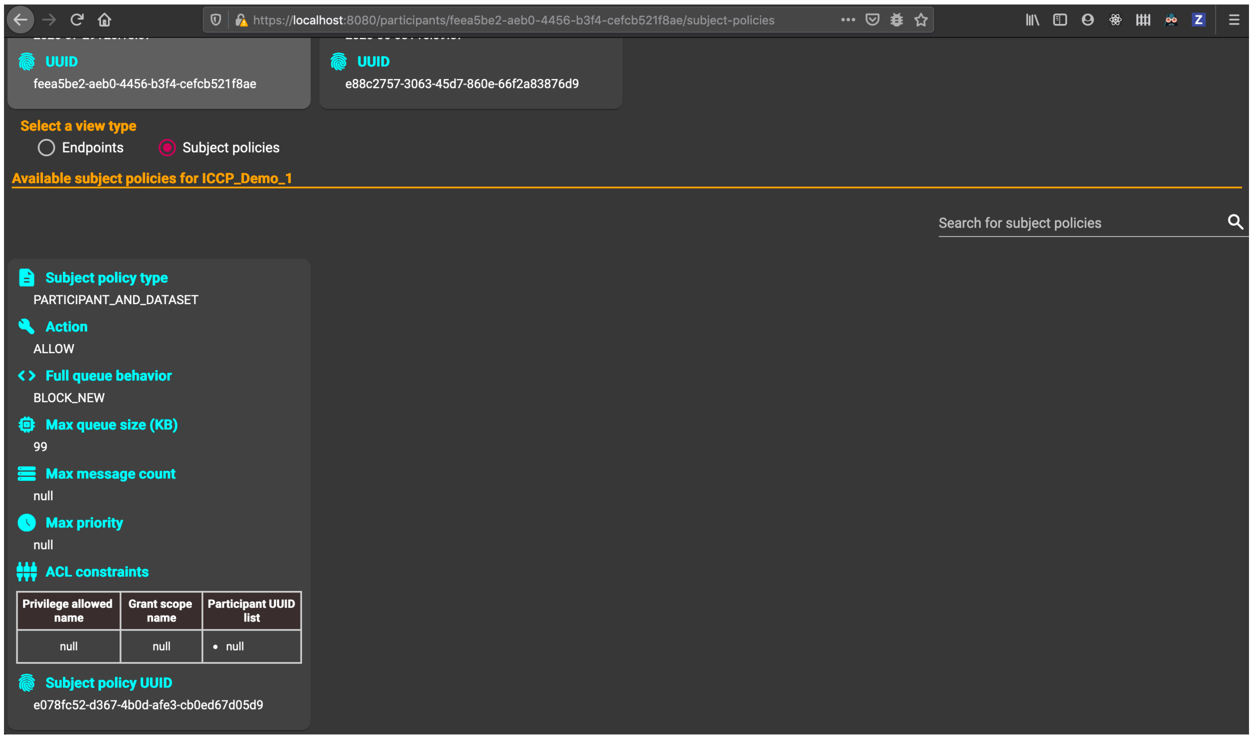
Select a view type
[✓ Endpoints] Subject policies

Available endpoints for ICCP_Demo_1

Search for endpoints

Username
app_rt_1
Description
ICCP Demo EP-Consumer (ADMIN)
Created on
2020-06-03T14:16:40
UUID
4b3b819e-94bd-4adf-b461-17ccb58ac870
Participant UUID
feea5be2-aeb0-4456-b3f4-cefcb521f8ae

Figure 15. Web UI Endpoints List



https://localhost:8080/participants/feea5be2-aeb0-4456-b3f4-cefcb521f8ae/subject-policies

UUID
feea5be2-aeb0-4456-b3f4-cefcb521f8ae

UUID
e88c2757-3063-45d7-860e-66f2a83876d9

Select a view type
Endpoints [✓ Subject policies]

Available subject policies for ICCP_Demo_1

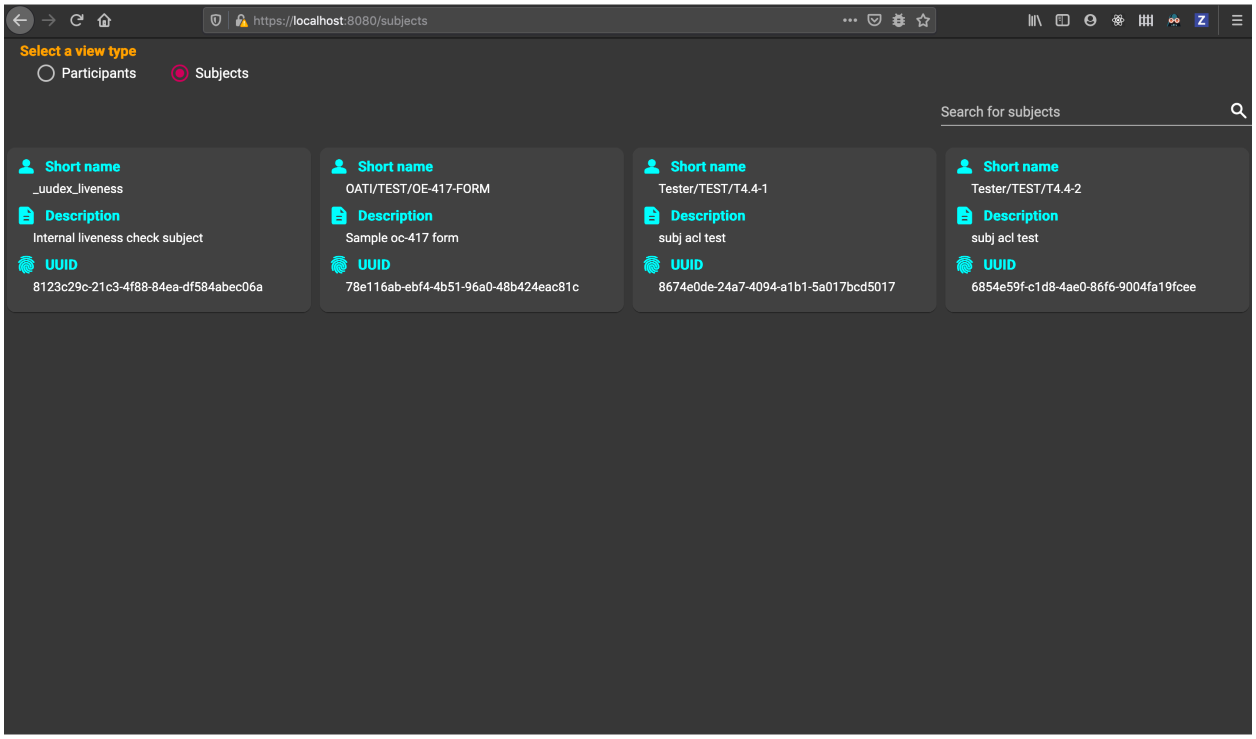
Search for subject policies

Subject policy type
PARTICIPANT_AND_DATASET
Action
ALLOW
Full queue behavior
BLOCK_NEW
Max queue size (KB)
99
Max message count
null
Max priority
null
ACL constraints

Privilege allowed name	Grant scope name	Participant UUID list
null	null	• null

Subject policy UUID
e078fc52-d367-4b0d-afe3-cb0ed67d05d9

Figure 16. Web UI Subject Policy View


https://localhost:8080/subjects

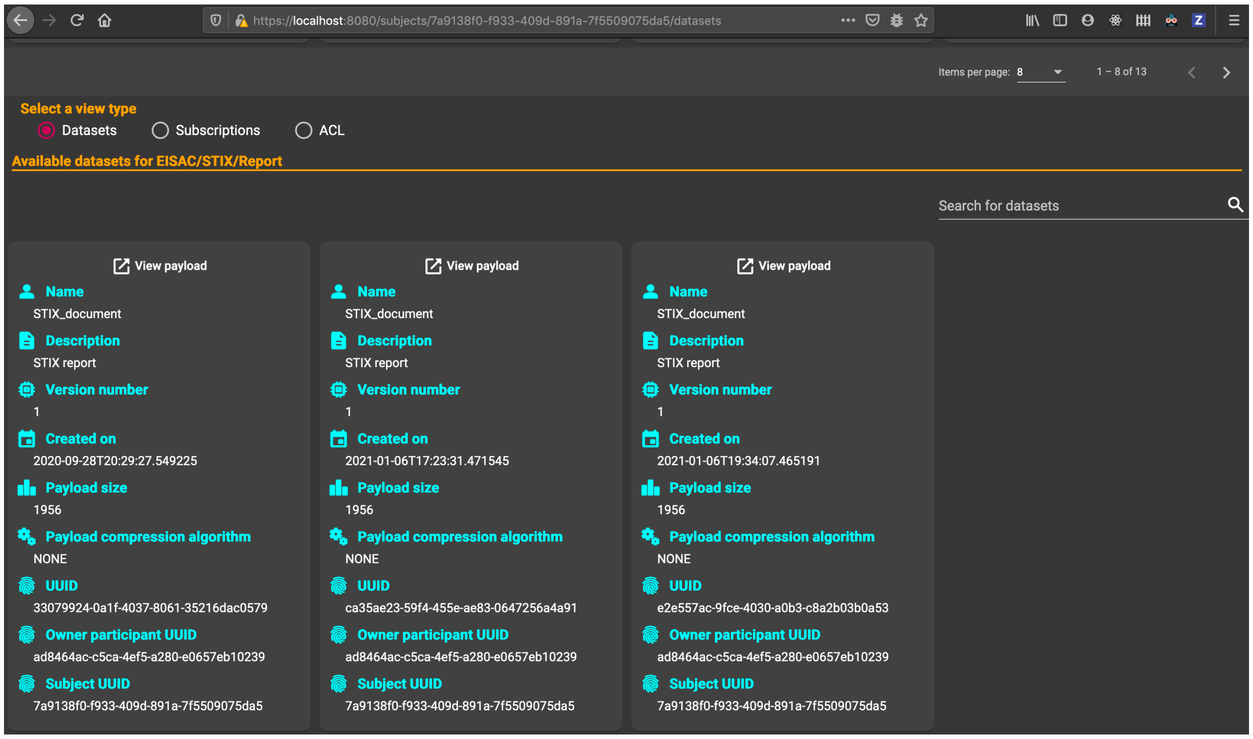
Select a view type

Participants [✓ Subjects]

Search for subjects

Short name	Description	UUID
_uudex_liveness	Internal liveness check subject	8123c29c-21c3-4f88-84ea-df584abec06a
OATI/TEST/OE-417-FORM	Sample oc-417 form	78e116ab-ebf4-4b51-96a0-48b424eac81c
Tester/TEST/T4.4-1	subj acl test	8674e0de-24a7-4094-a1b1-5a017bcd5017
Tester/TEST/T4.4-2	subj acl test	6854e59f-c1d8-4ae0-86f6-9004fa19fcee

Figure 17. Web UI Subject List


https://localhost:8080/subjects/7a9138f0-f933-409d-891a-7f5509075da5/datasets

Items per page: 8 1 – 8 of 13

Select a view type

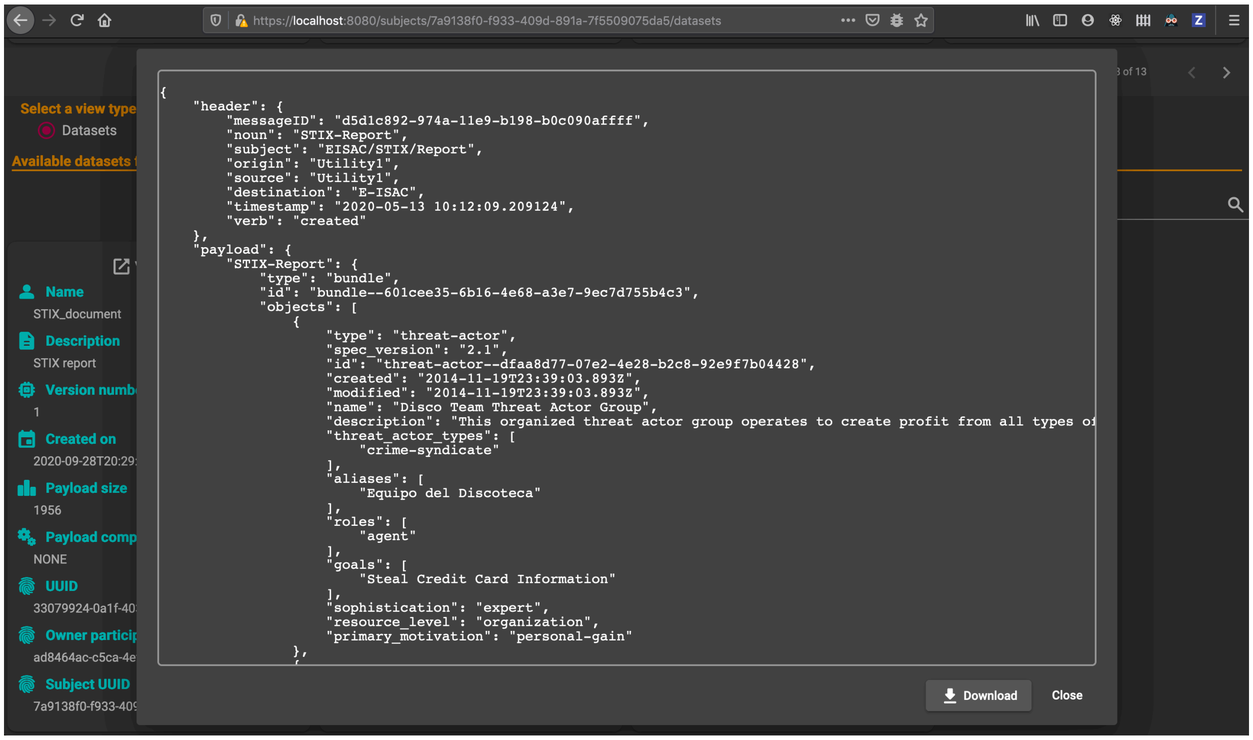
[✓ Datasets] Subscriptions ACL

Available datasets for EISAC/STIX/Report

Search for datasets

Name	STIX_document	STIX_document	STIX_document
Description	STIX report	STIX report	STIX report
Version number	1	1	1
Created on	2020-09-28T20:29:27.549225	2021-01-06T17:23:31.471545	2021-01-06T19:34:07.465191
Payload size	1956	1956	1956
Payload compression algorithm	NONE	NONE	NONE
UUID	33079924-0a1f-4037-8061-35216dac0579	ca35ae23-59f4-455e-ae83-0647256a4a91	e2e557ac-9fce-4030-a0b3-c8a2b03b0a53
Owner participant UUID	ad8464ac-c5ca-4ef5-a280-e0657eb10239	ad8464ac-c5ca-4ef5-a280-e0657eb10239	ad8464ac-c5ca-4ef5-a280-e0657eb10239
Subject UUID	7a9138f0-f933-409d-891a-7f5509075da5	7a9138f0-f933-409d-891a-7f5509075da5	7a9138f0-f933-409d-891a-7f5509075da5

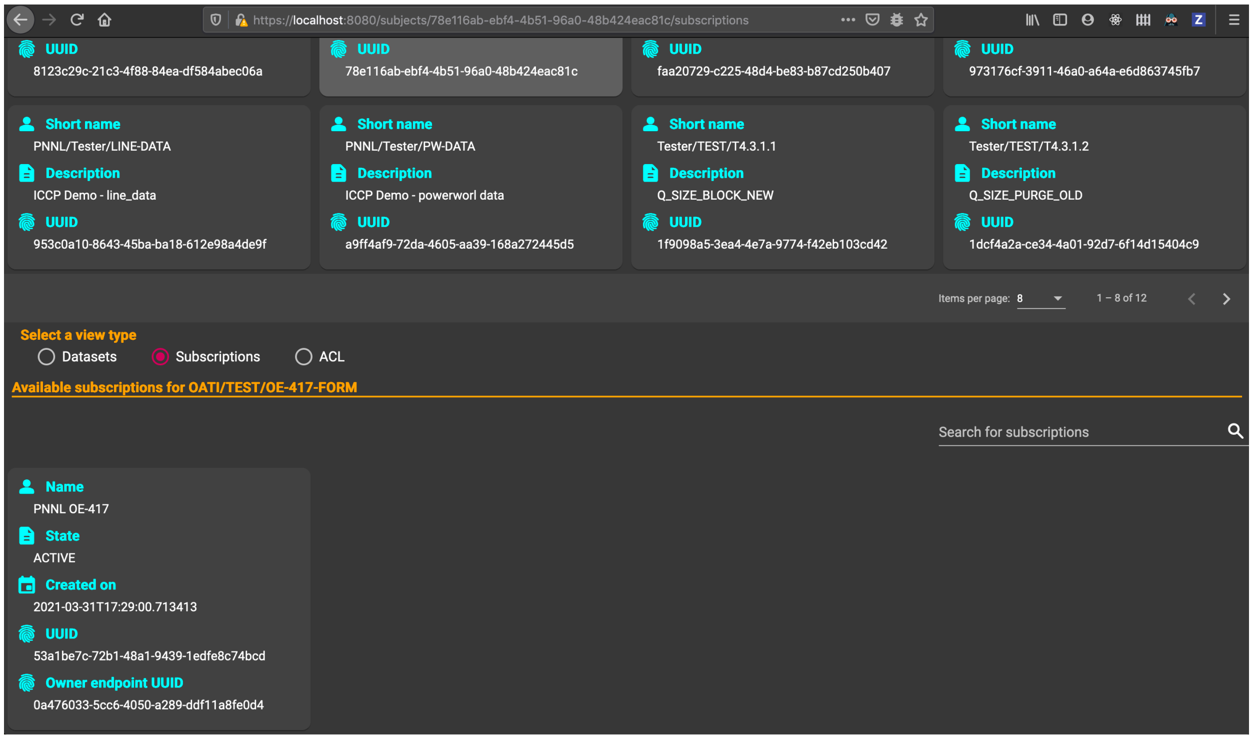
Figure 18. Web UI Data Sets List


https://localhost:8080/subjects/7a9138f0-f933-409d-891a-7f5509075da5/datasets
3 of 13
Select a view type
Datasets
Available datasets f
Download
Close

Name
STIX_document
Description
STIX report
Version numb
1
Created on
2020-09-28T20:29
Payload size
1956
Payload comp
NONE
UUID
33079924-0a1f-40
Owner particip
ad8464ac-c5ca-4e
Subject UUID
7a9138f0-f933-409

```
{
    "header": {
        "messageID": "d5d1c892-974a-11e9-b198-b0c090affff",
        "noun": "STIX-Report",
        "subject": "EISAC/STIX/Report",
        "origin": "Utility1",
        "source": "Utility1",
        "destination": "E-ISAC",
        "timestamp": "2020-05-13 10:12:09.209124",
        "verb": "created"
    },
    "payload": {
        "STIX-Report": {
            "type": "bundle",
            "id": "bundle--601cee35-6b16-4e68-a3e7-9ec7d755b4c3",
            "objects": [
                {
                    "type": "threat-actor",
                    "spec_version": "2.1",
                    "id": "threat-actor--dfaa8d77-07e2-4e28-b2c8-92e9f7b04428",
                    "created": "2014-11-19T23:39:03.893Z",
                    "modified": "2014-11-19T23:39:03.893Z",
                    "name": "Disco Team Threat Actor Group",
                    "description": "This organized threat actor group operates to create profit from all types of
                    "threat_actor_types": [
                        "crime-syndicate"
                    ],
                    "aliases": [
                        "Equipo del Discoteca"
                    ],
                    "roles": [
                        "agent"
                    ],
                    "goals": [
                        "Steal Credit Card Information"
                    ],
                    "sophistication": "expert",
                    "resource_level": "organization",
                    "primary_motivation": "personal-gain"
                },
```

Figure 19. Web UI Payload Inspection


https://localhost:8080/subjects/78e116ab-ebf4-4b51-96a0-48b424eac81c/subscriptions
Items per page: 8
1 – 8 of 12
Select a view type
Search for subscriptions

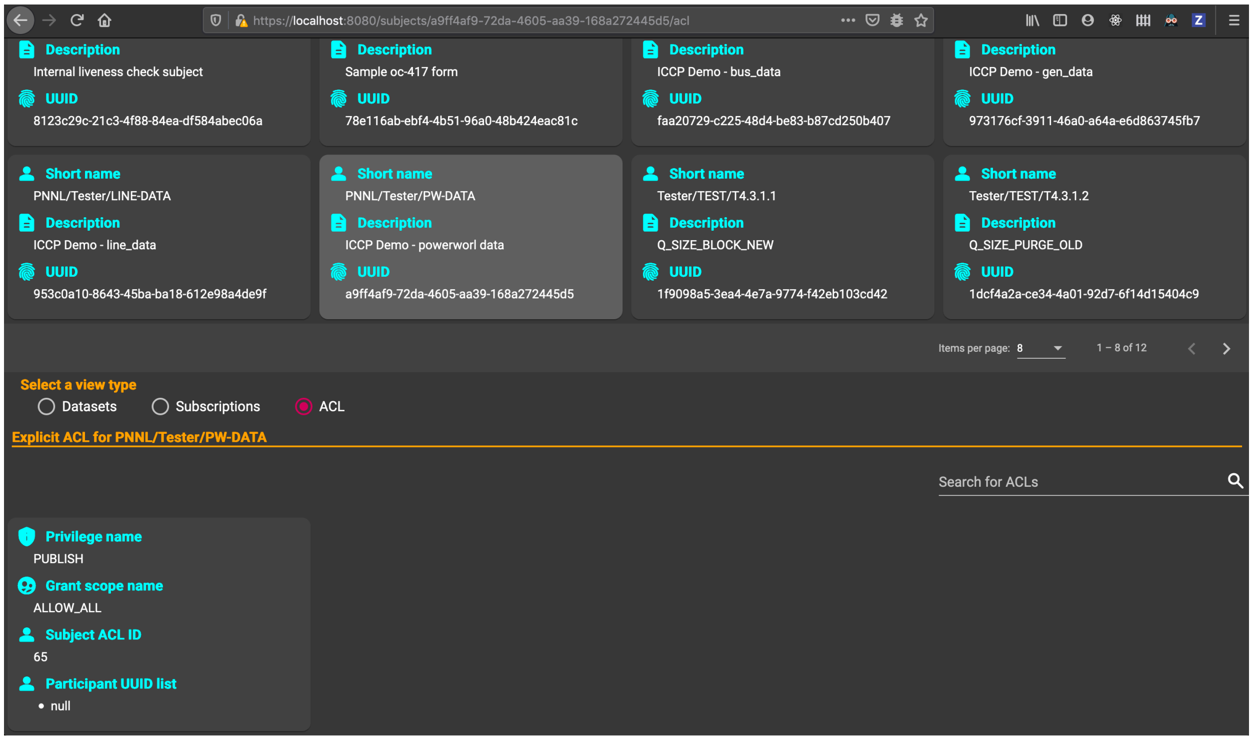
UUID 8123c29c-21c3-4f88-84ea-df584abec06a	UUID 78e116ab-ebf4-4b51-96a0-48b424eac81c	UUID faa20729-c225-48d4-be83-b87cd250b407	UUID 973176cf-3911-46a0-a64a-e6d863745fb7
Short name PNNL/Tester/LINE-DATA Description ICCP Demo - line_data UUID 953c0a10-8643-45ba-ba18-612e98a4de9f	Short name PNNL/Tester/PW-DATA Description ICCP Demo - powerworl data UUID a9ff4af9-72da-4605-aa39-168a272445d5	Short name Tester/TEST/T4.3.1.1 Description Q_SIZE_BLOCK_NEW UUID 1f9098a5-3ea4-4e7a-9774-f42eb103cd42	Short name Tester/TEST/T4.3.1.2 Description Q_SIZE_PURGE_OLD UUID 1dcf4a2a-ce34-4a01-92d7-6f14d15404c9

Datasets [✓ Subscriptions] ACL

Available subscriptions for OATI/TEST/OE-417-FORM

Name
PNNL OE-417
State
ACTIVE
Created on
2021-03-31T17:29:00.713413
UUID
53a1be7c-72b1-48a1-9439-1edfe8c74bcd
Owner endpoint UUID
0a476033-5cc6-4050-a289-ddf11a8fe0d4

Figure 20. Web UI Subscriptions List



https://localhost:8080/subjects/a9ff4af9-72da-4605-aa39-168a272445d5/acl

Description
Internal liveness check subject
UUID
8123c29c-21c3-4f88-84ea-df584abec06a

Description
Sample oc-417 form
UUID
78e116ab-ebf4-4b51-96a0-48b424eac81c

Description
ICCP Demo - bus_data
UUID
faa20729-c225-48d4-be83-b87cd250b407

Description
ICCP Demo - gen_data
UUID
973176cf-3911-46a0-a64a-e6d863745fb7

Short name
PNNL/Tester/LINE-DATA
Description
ICCP Demo - line_data
UUID
953c0a10-8643-45ba-ba18-612e98a4de9f

Short name
PNNL/Tester/PW-DATA
Description
ICCP Demo - powerworl data
UUID
a9ff4af9-72da-4605-aa39-168a272445d5

Short name
Tester/TEST/T4.3.1.1
Description
Q_SIZE_BLOCK_NEW
UUID
1f9098a5-3ea4-4e7a-9774-f42eb103cd42

Short name
Tester/TEST/T4.3.1.2
Description
Q_SIZE_PURGE_OLD
UUID
1dcf4a2a-ce34-4a01-92d7-6f14d15404c9

Items per page: 8 1 – 8 of 12

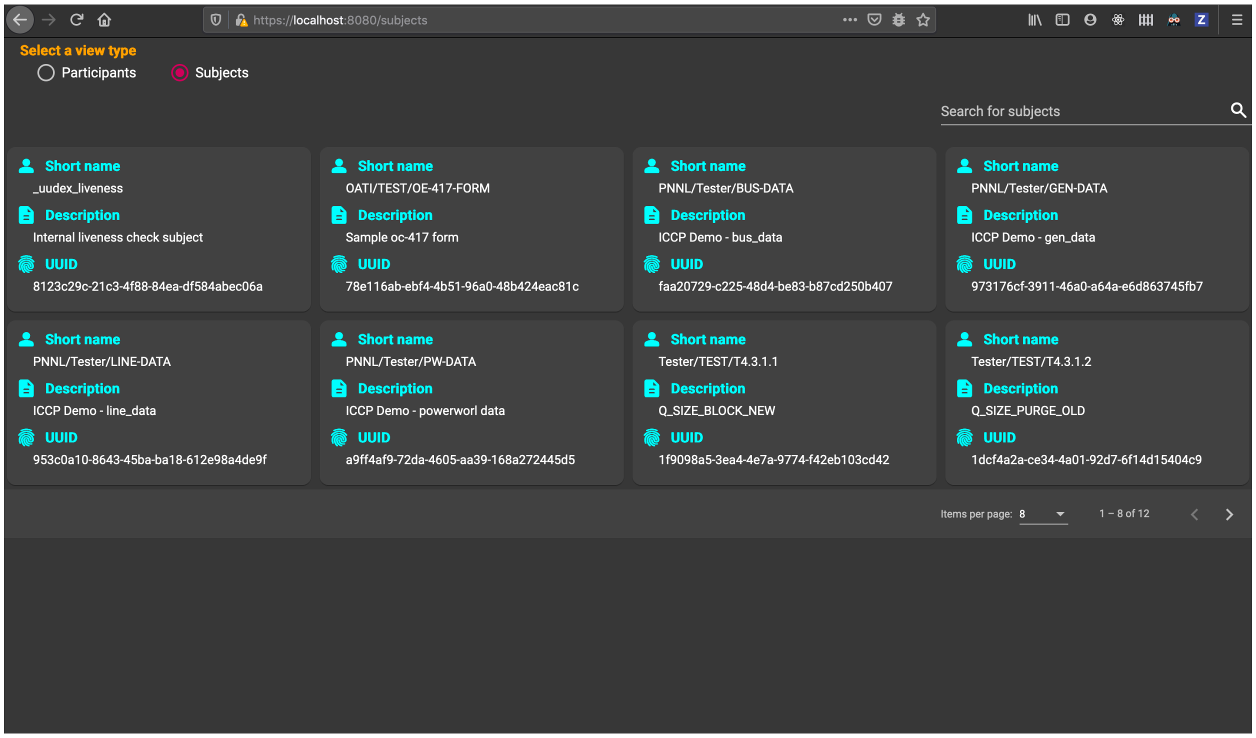
Select a view type
Datasets Subscriptions [✓ ACL]

Explicit ACL for PNNL/Tester/PW-DATA

Search for ACLs

Privilege name
PUBLISH
Grant scope name
ALLOW_ALL
Subject ACL ID
65
Participant UUID list
- null

Figure 21. Web UI ACL View



https://localhost:8080/subjects

Select a view type
Participants [✓ Subjects]

Search for subjects

Short name
_uudex_liveness
Description
Internal liveness check subject
UUID
8123c29c-21c3-4f88-84ea-df584abec06a

Short name
OATI/TEST/OE-417-FORM
Description
Sample oc-417 form
UUID
78e116ab-ebf4-4b51-96a0-48b424eac81c

Short name
PNNL/Tester/BUS-DATA
Description
ICCP Demo - bus_data
UUID
faa20729-c225-48d4-be83-b87cd250b407

Short name
PNNL/Tester/GEN-DATA
Description
ICCP Demo - gen_data
UUID
973176cf-3911-46a0-a64a-e6d863745fb7

Short name
PNNL/Tester/LINE-DATA
Description
ICCP Demo - line_data
UUID
953c0a10-8643-45ba-ba18-612e98a4de9f

Short name
PNNL/Tester/PW-DATA
Description
ICCP Demo - powerworl data
UUID
a9ff4af9-72da-4605-aa39-168a272445d5

Short name
Tester/TEST/T4.3.1.1
Description
Q_SIZE_BLOCK_NEW
UUID
1f9098a5-3ea4-4e7a-9774-f42eb103cd42

Short name
Tester/TEST/T4.3.1.2
Description
Q_SIZE_PURGE_OLD
UUID
1dcf4a2a-ce34-4a01-92d7-6f14d15404c9

Items per page: 8 1 – 8 of 12

Figure 22. Web UI Administrator Subjects View

4.0 Conclusions and Next Steps

In general, all the demonstration tasks performed were successful. As noted, the target of one demonstration needed to be modified to allow it to be executed.

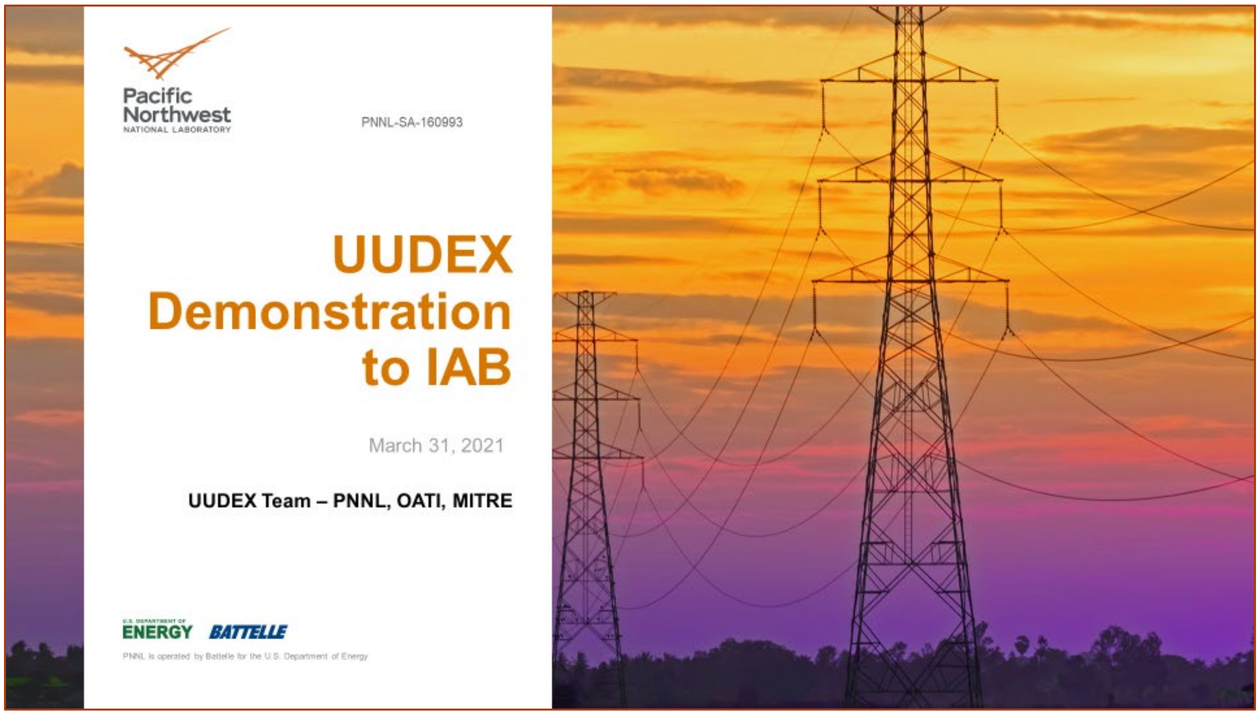
The project team also shared the next steps for the project with the IAB. These include:

- Finish existing documentation assignments (primarily with Security topics)
  - Re-thinking of Subject Creation Policy and associated processing, including Group and Role concepts
  - Certificate Authority Hierarchy
  - We are interested in IAB input on some of these topics – will follow up with an email
- Integrate Performance Monitoring code developed but not included in demonstration
  - Shows overall health status and round-trip communication performance
- Clean up project documentation based on design evolution and implementation
- Looking for opportunities to field test UUDEX at utility (or other) sites
- Propose UUDEX to the IEEE for standardization
  - Process started
  - Looking for statements of support to assist the standardization kickoff process

A brief question and answer session with the IAB concluded the demonstration.

Appendix A Demonstration Presentation

The following presentation was used to guide the demonstration, and to introduce each demonstration task.



Pacific Northwest NATIONAL LABORATORY

PNNL-SA-160993

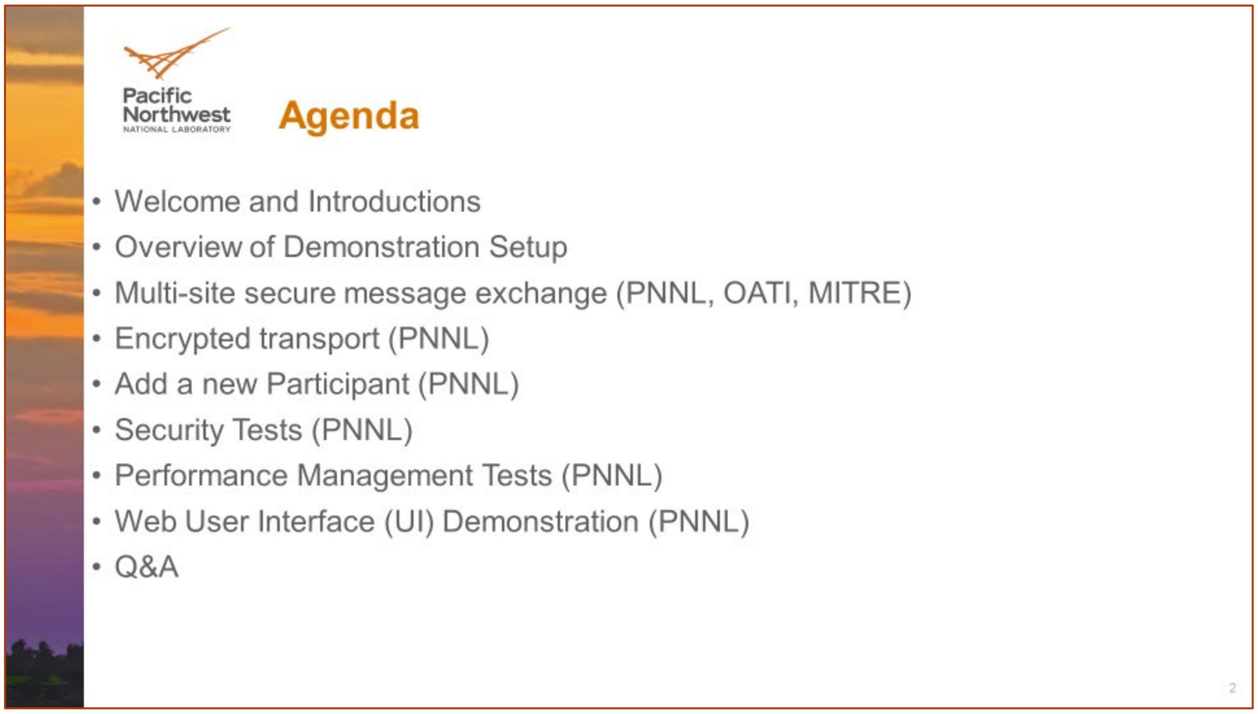
UUDEX Demonstration to IAB

March 31, 2021

UUDEX Team – PNNL, OATI, MITRE

U.S. DEPARTMENT OF ENERGY BATTELLE

PNNL is operated by Battelle for the U.S. Department of Energy

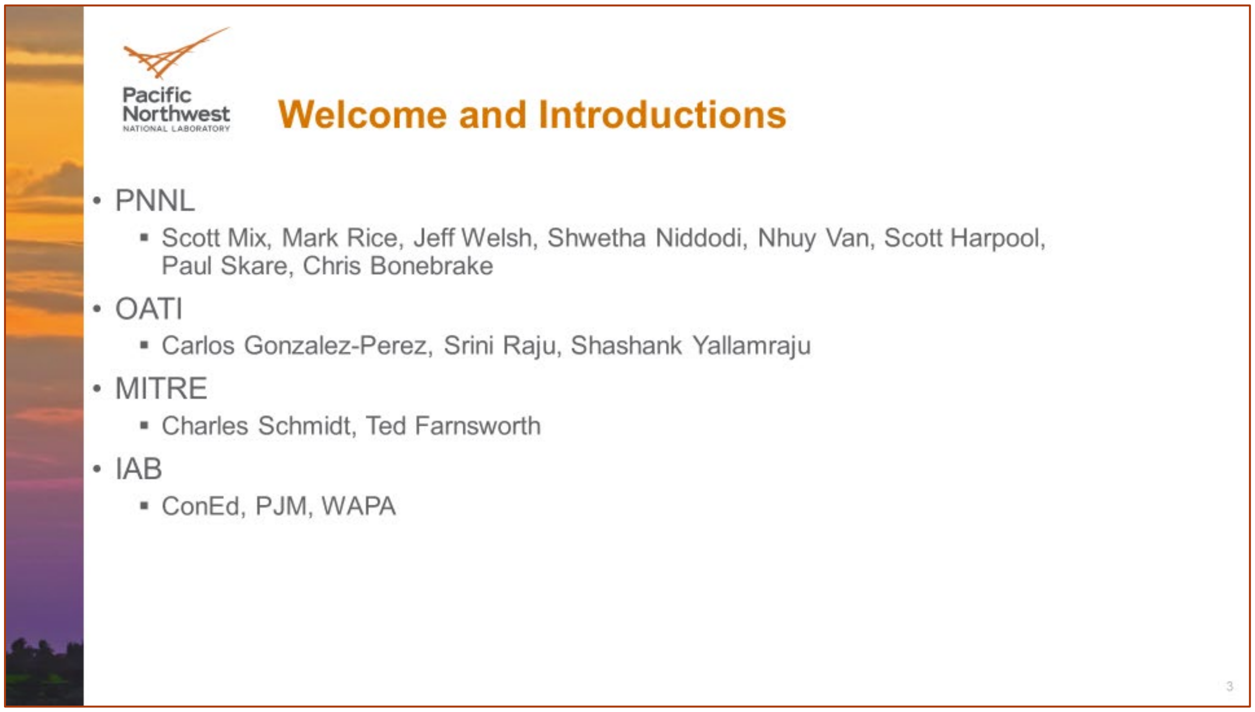


Pacific Northwest NATIONAL LABORATORY

Agenda

- Welcome and Introductions
- Overview of Demonstration Setup
- Multi-site secure message exchange (PNNL, OATI, MITRE)
- Encrypted transport (PNNL)
- Add a new Participant (PNNL)
- Security Tests (PNNL)
- Performance Management Tests (PNNL)
- Web User Interface (UI) Demonstration (PNNL)
- Q&A

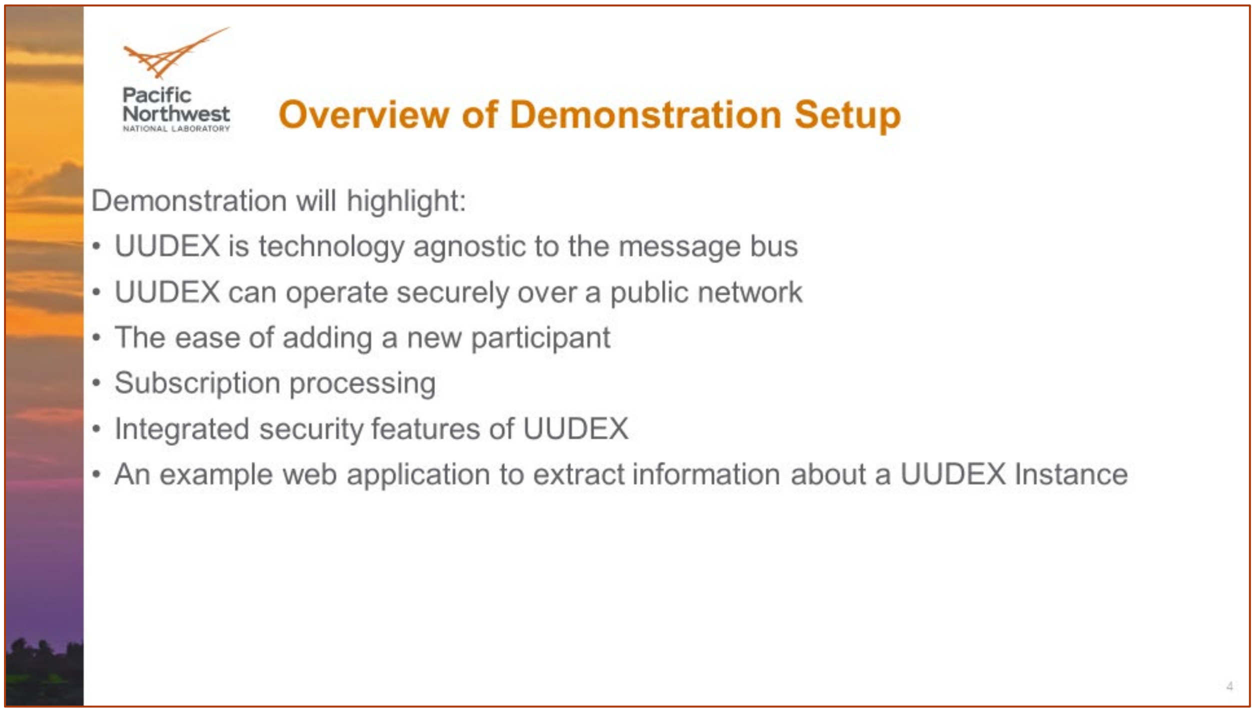
2



Welcome and Introductions

- PNNL
  - Scott Mix, Mark Rice, Jeff Welsh, Shwetha Niddodi, Nhuy Van, Scott Harpool, Paul Skare, Chris Bonebrake
- OATI
  - Carlos Gonzalez-Perez, Srini Raju, Shashank Yallamraju
- MITRE
  - Charles Schmidt, Ted Farnsworth
- IAB
  - ConEd, PJM, WAPA

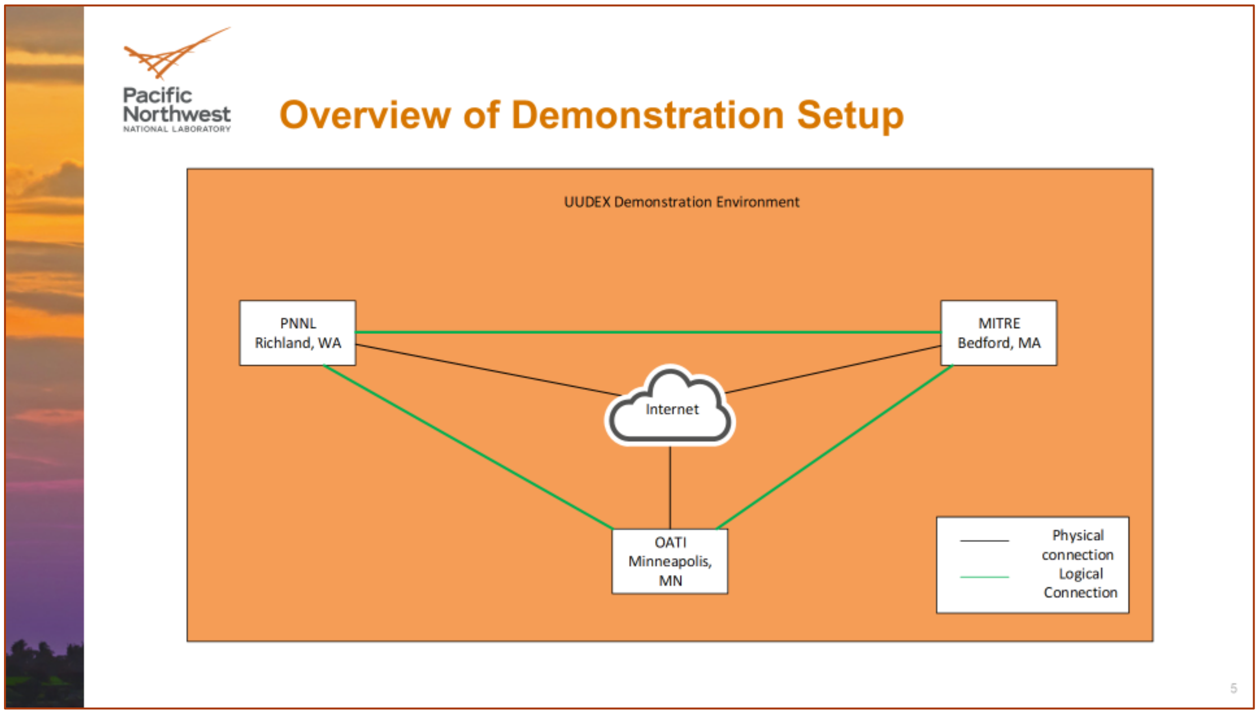
3



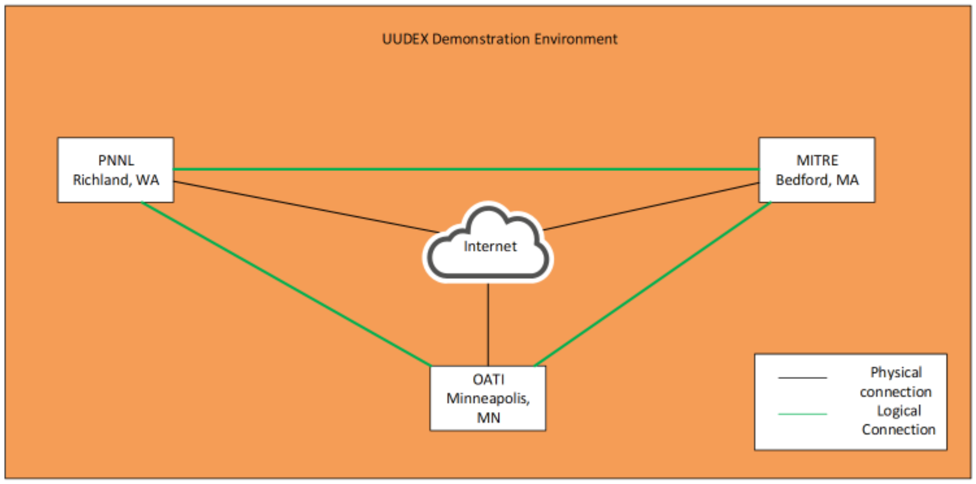
Overview of Demonstration Setup

Demonstration will highlight:
- UUDEX is technology agnostic to the message bus
- UUDEX can operate securely over a public network
- The ease of adding a new participant
- Subscription processing
- Integrated security features of UUDEX
- An example web application to extract information about a UUDEX Instance

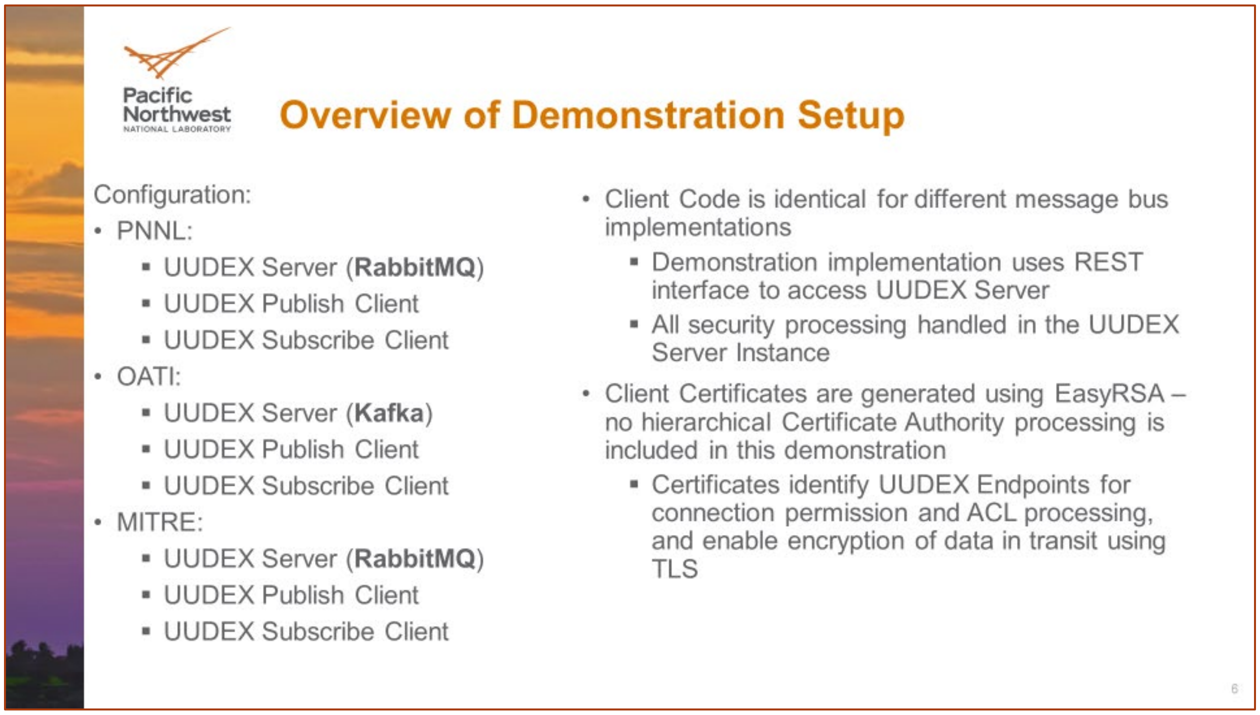
4



Overview of Demonstration Setup



5



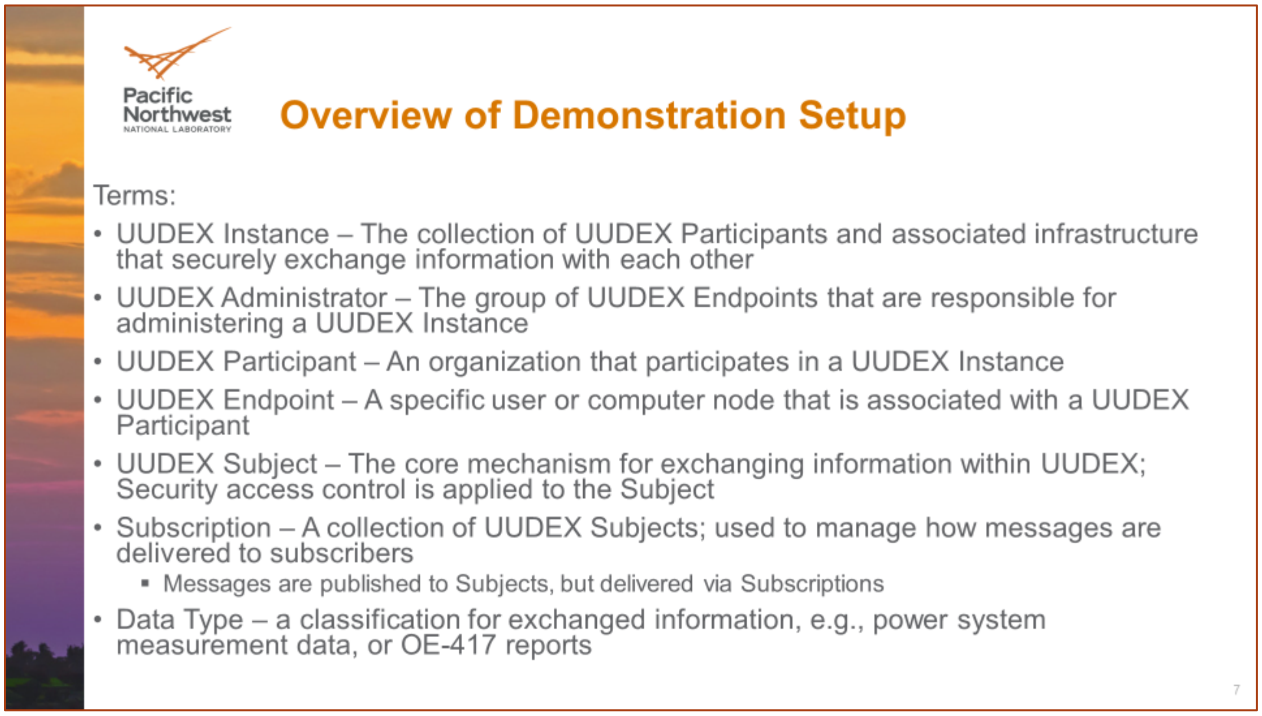
Overview of Demonstration Setup

Configuration:

- PNNL:
  - UUDEX Server (**RabbitMQ**)
  - UUDEX Publish Client
  - UUDEX Subscribe Client
- OATI:
  - UUDEX Server (**Kafka**)
  - UUDEX Publish Client
  - UUDEX Subscribe Client
- MITRE:
  - UUDEX Server (**RabbitMQ**)
  - UUDEX Publish Client
  - UUDEX Subscribe Client

- Client Code is identical for different message bus implementations
  - Demonstration implementation uses REST interface to access UUDEX Server
  - All security processing handled in the UUDEX Server Instance
- Client Certificates are generated using EasyRSA – no hierarchical Certificate Authority processing is included in this demonstration
  - Certificates identify UUDEX Endpoints for connection permission and ACL processing, and enable encryption of data in transit using TLS

6

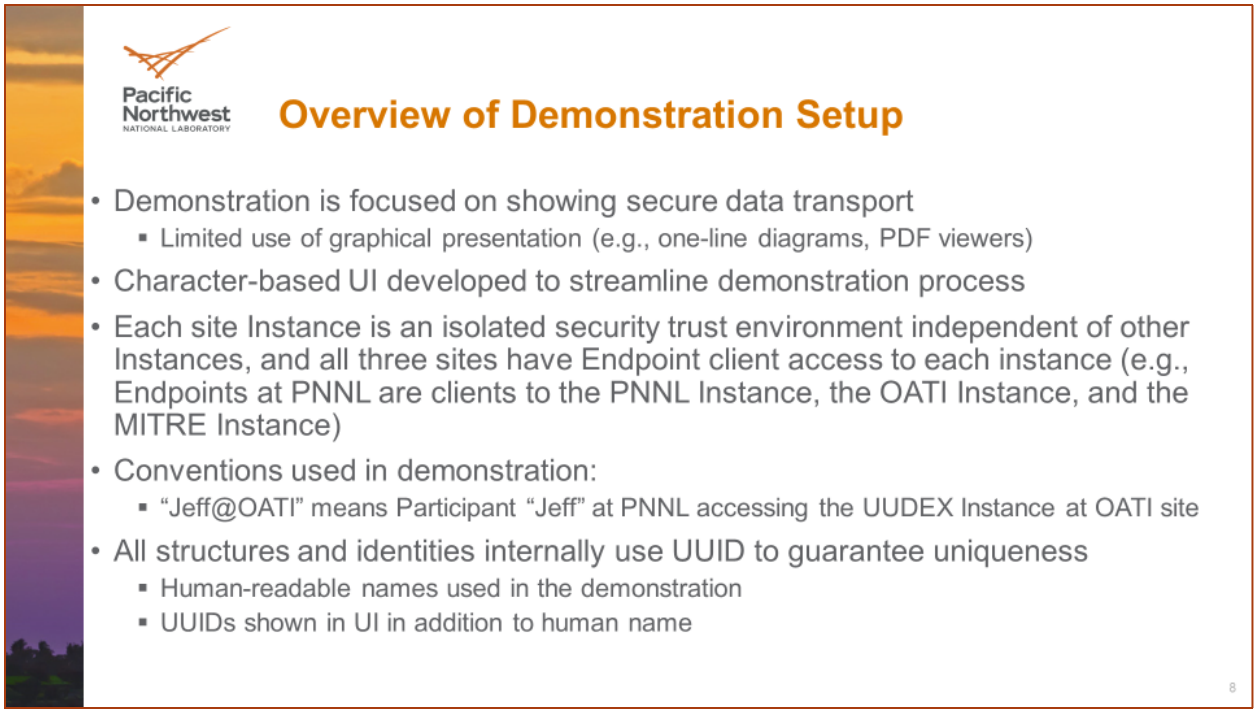


Overview of Demonstration Setup

Terms:

- UUDEX Instance – The collection of UUDEX Participants and associated infrastructure that securely exchange information with each other
- UUDEX Administrator – The group of UUDEX Endpoints that are responsible for administering a UUDEX Instance
- UUDEX Participant – An organization that participates in a UUDEX Instance
- UUDEX Endpoint – A specific user or computer node that is associated with a UUDEX Participant
- UUDEX Subject – The core mechanism for exchanging information within UUDEX; Security access control is applied to the Subject
- Subscription – A collection of UUDEX Subjects; used to manage how messages are delivered to subscribers
  - Messages are published to Subjects, but delivered via Subscriptions
- Data Type – a classification for exchanged information, e.g., power system measurement data, or OE-417 reports

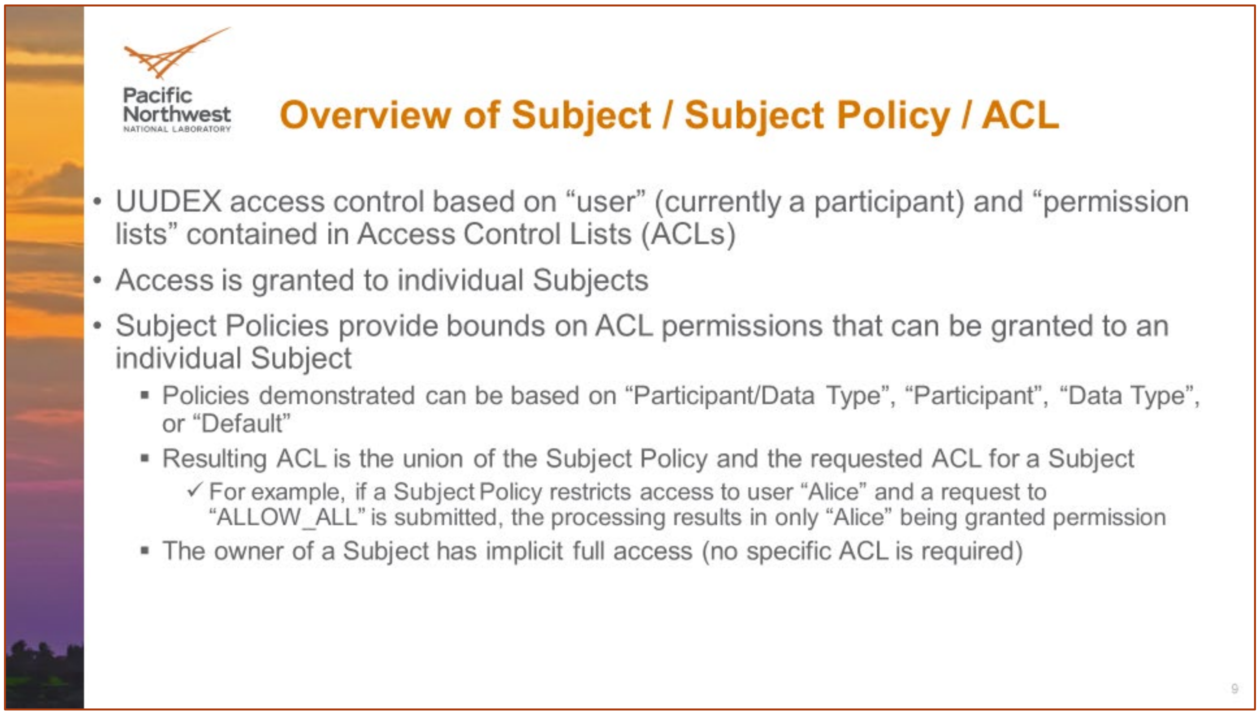
7



Overview of Demonstration Setup

- Demonstration is focused on showing secure data transport
  - Limited use of graphical presentation (e.g., one-line diagrams, PDF viewers)
- Character-based UI developed to streamline demonstration process
- Each site Instance is an isolated security trust environment independent of other Instances, and all three sites have Endpoint client access to each instance (e.g., Endpoints at PNNL are clients to the PNNL Instance, the OATI Instance, and the MITRE Instance)
- Conventions used in demonstration:
  - “Jeff@OATI” means Participant “Jeff” at PNNL accessing the UUDEX Instance at OATI site
- All structures and identities internally use UUID to guarantee uniqueness
  - Human-readable names used in the demonstration
  - UUIDs shown in UI in addition to human name

8

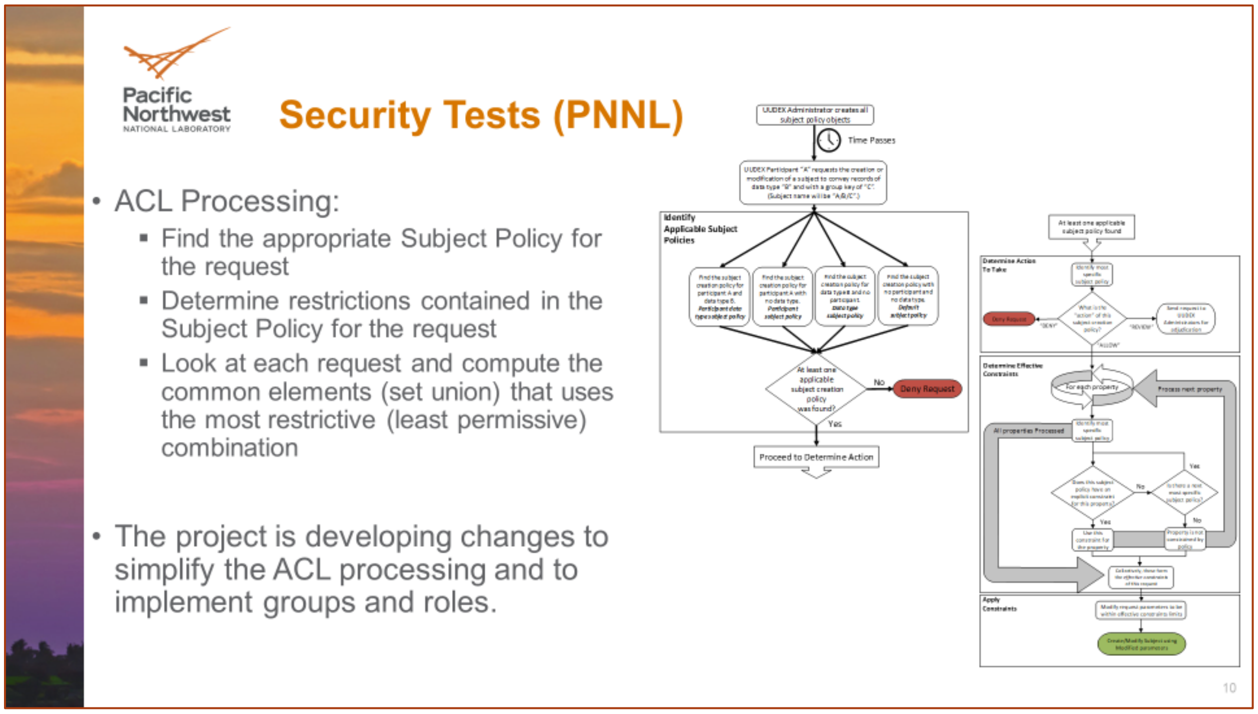


Pacific Northwest NATIONAL LABORATORY

Overview of Subject / Subject Policy / ACL

- UUDEX access control based on “user” (currently a participant) and “permission lists” contained in Access Control Lists (ACLs)
- Access is granted to individual Subjects
- Subject Policies provide bounds on ACL permissions that can be granted to an individual Subject
  - Policies demonstrated can be based on “Participant/Data Type”, “Participant”, “Data Type”, or “Default”
  - Resulting ACL is the union of the Subject Policy and the requested ACL for a Subject
    - ✓ For example, if a Subject Policy restricts access to user “Alice” and a request to “ALLOW_ALL” is submitted, the processing results in only “Alice” being granted permission
  - The owner of a Subject has implicit full access (no specific ACL is required)

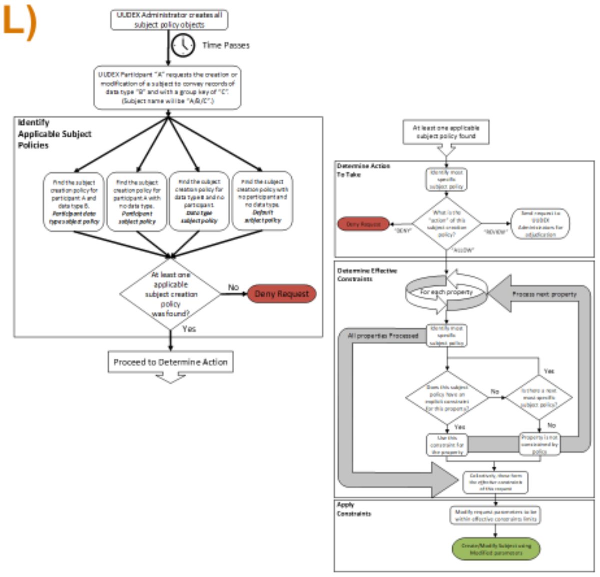
9



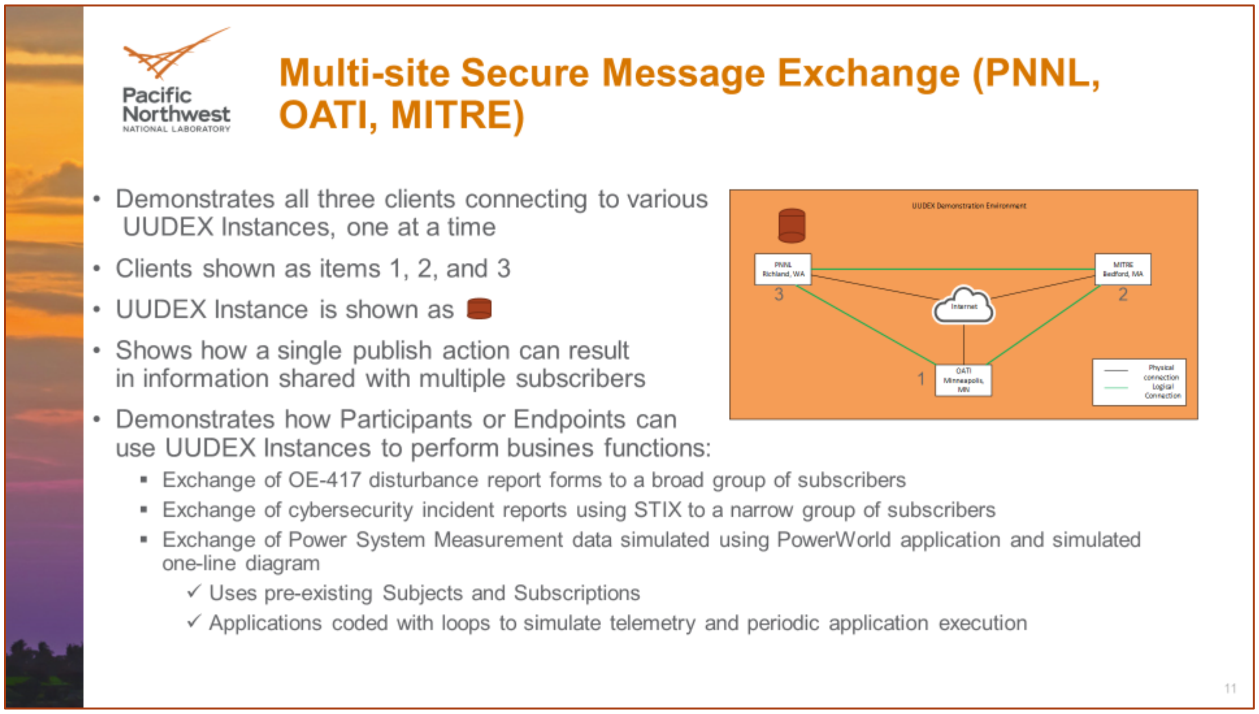
Pacific Northwest NATIONAL LABORATORY

Security Tests (PNNL)

- ACL Processing:
  - Find the appropriate Subject Policy for the request
  - Determine restrictions contained in the Subject Policy for the request
  - Look at each request and compute the common elements (set union) that uses the most restrictive (least permissive) combination
- The project is developing changes to simplify the ACL processing and to implement groups and roles.


UUDEX Administrator creates all subject policy objects
Time Passes
UUDEX Participant "A" requests the creation or modification of a subject to convey records of data type "B" and with a group key of "C". (Subject name will be "A/B/C".)
Identify Applicable Subject Policies
Find the subject creation policy for participant A and data type B. Participant data type subject policy
Find the subject creation policy for participant A with no data type. Participant subject policy
Find the subject creation policy for data type B and no participant. Data type subject policy
Find the subject creation policy with no participant and no data type. Default subject policy
At least one applicable subject creation policy was found?
No
Deny Request
Yes
Proceed to Determine Action
At least one applicable subject policy found
Determine Action To Take
Identify most specific subject policy
What is the "action" of this subject creation policy?
"DENY"
Deny Request
"REVIEW"
Send request to UUDEX Administrators for adjudication
"ALLOW"
Determine Effective Constraints
For each property
Process next property
All properties Processed
Identify most specific subject policy
Does this subject policy have an explicit constraint for this property?
No
Is there a next most specific subject policy?
Yes
No
Yes
Use this constraint for the property
Property is not constrained by policy
Collectively, these form the effective constraints of this request
Apply Constraints
Modify request parameters to be within effective constraints limits
Create/Modify Subject using Modified parameters

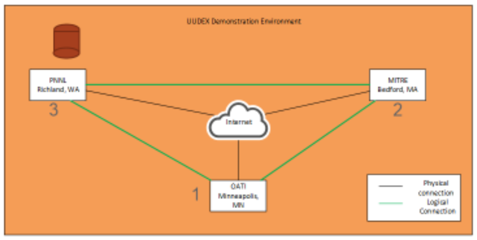
10



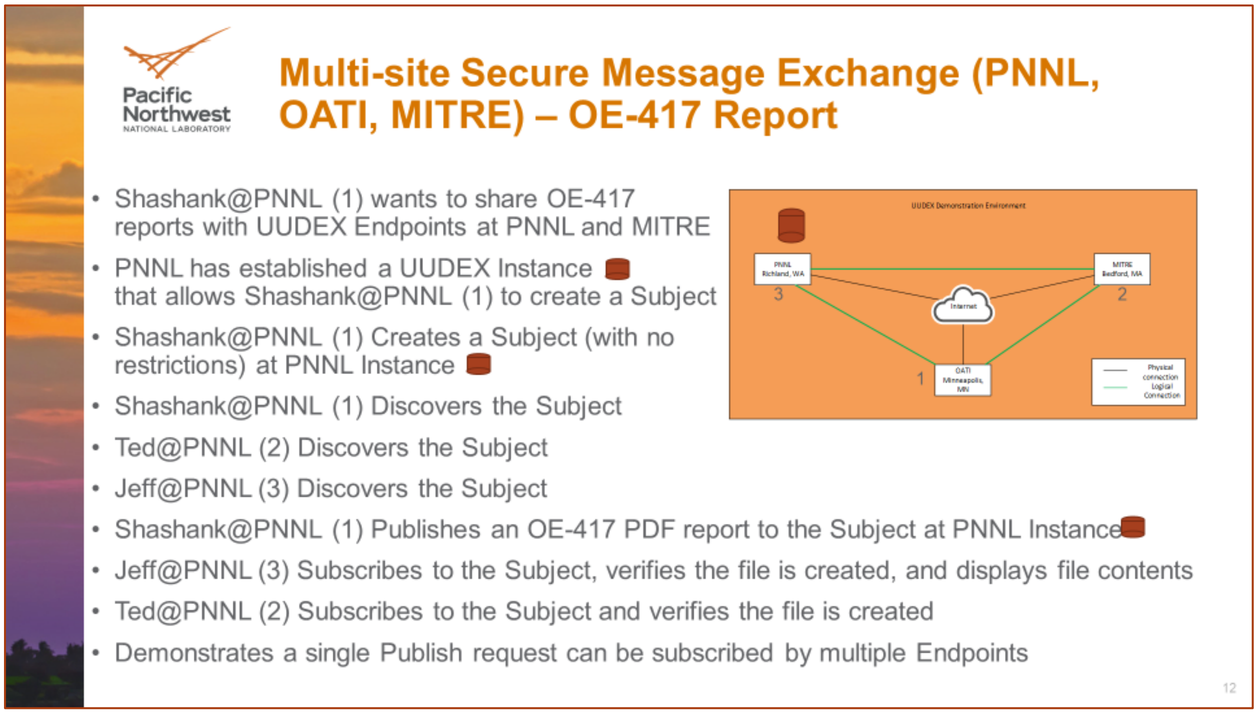
Pacific Northwest NATIONAL LABORATORY

Multi-site Secure Message Exchange (PNNL, OATI, MITRE)

- Demonstrates all three clients connecting to various UUDEX Instances, one at a time
- Clients shown as items 1, 2, and 3
- UUDEX Instance is shown as
- Shows how a single publish action can result in information shared with multiple subscribers
- Demonstrates how Participants or Endpoints can use UUDEX Instances to perform busines functions:
  - Exchange of OE-417 disturbance report forms to a broad group of subscribers
  - Exchange of cybersecurity incident reports using STIX to a narrow group of subscribers
  - Exchange of Power System Measurement data simulated using PowerWorld application and simulated one-line diagram
    - ✓ Uses pre-existing Subjects and Subscriptions
    - ✓ Applications coded with loops to simulate telemetry and periodic application execution



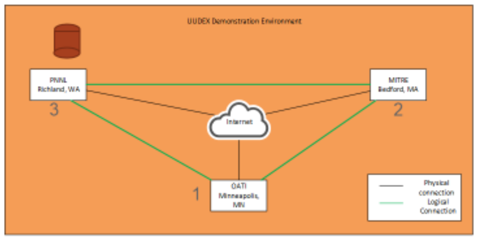
11



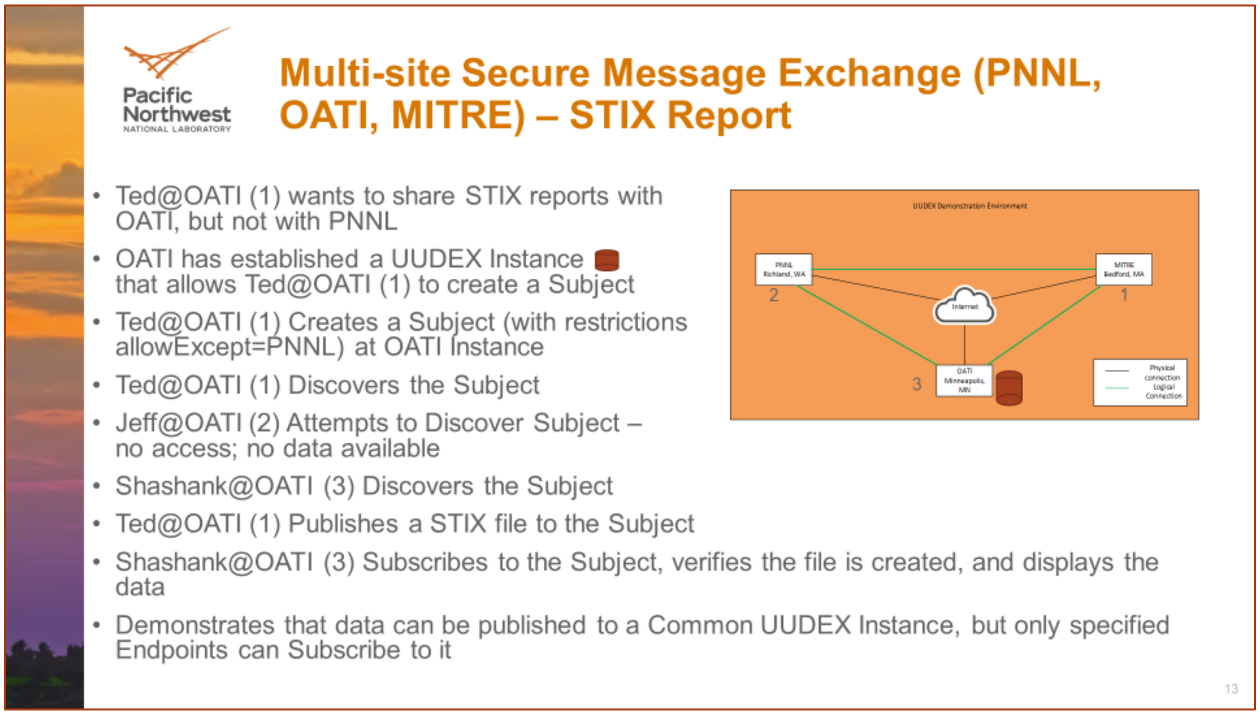
Pacific Northwest NATIONAL LABORATORY

Multi-site Secure Message Exchange (PNNL, OATI, MITRE) – OE-417 Report

- Shashank@PNNL (1) wants to share OE-417 reports with UUDEX Endpoints at PNNL and MITRE
- PNNL has established a UUDEX Instance that allows Shashank@PNNL (1) to create a Subject
- Shashank@PNNL (1) Creates a Subject (with no restrictions) at PNNL Instance
- Shashank@PNNL (1) Discovers the Subject
- Ted@PNNL (2) Discovers the Subject
- Jeff@PNNL (3) Discovers the Subject
- Shashank@PNNL (1) Publishes an OE-417 PDF report to the Subject at PNNL Instance
- Jeff@PNNL (3) Subscribes to the Subject, verifies the file is created, and displays file contents
- Ted@PNNL (2) Subscribes to the Subject and verifies the file is created
- Demonstrates a single Publish request can be subscribed by multiple Endpoints

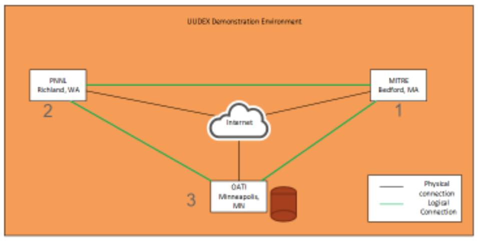


12

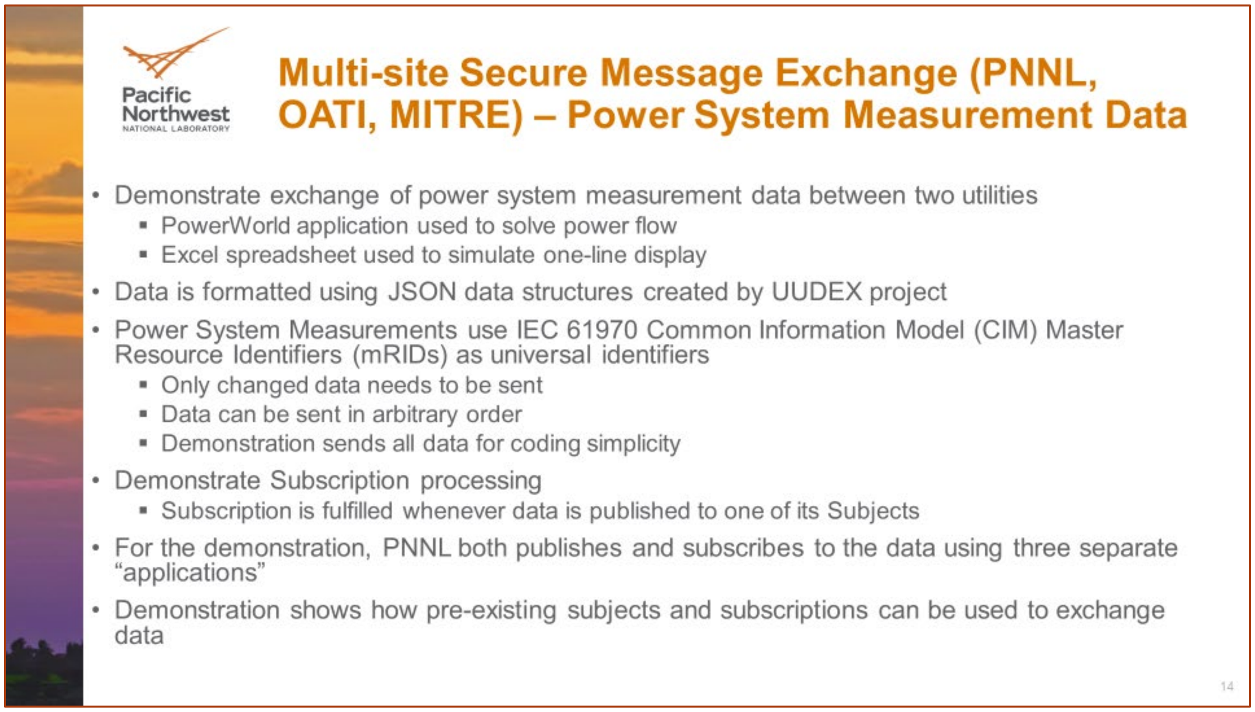


Multi-site Secure Message Exchange (PNNL, OATI, MITRE) – STIX Report

- Ted@OATI (1) wants to share STIX reports with OATI, but not with PNNL
- OATI has established a UUDEX Instance that allows Ted@OATI (1) to create a Subject
- Ted@OATI (1) Creates a Subject (with restrictions allowExcept=PNNL) at OATI Instance
- Ted@OATI (1) Discovers the Subject
- Jeff@OATI (2) Attempts to Discover Subject – no access; no data available
- Shashank@OATI (3) Discovers the Subject
- Ted@OATI (1) Publishes a STIX file to the Subject
- Shashank@OATI (3) Subscribes to the Subject, verifies the file is created, and displays the data
- Demonstrates that data can be published to a Common UUDEX Instance, but only specified Endpoints can Subscribe to it



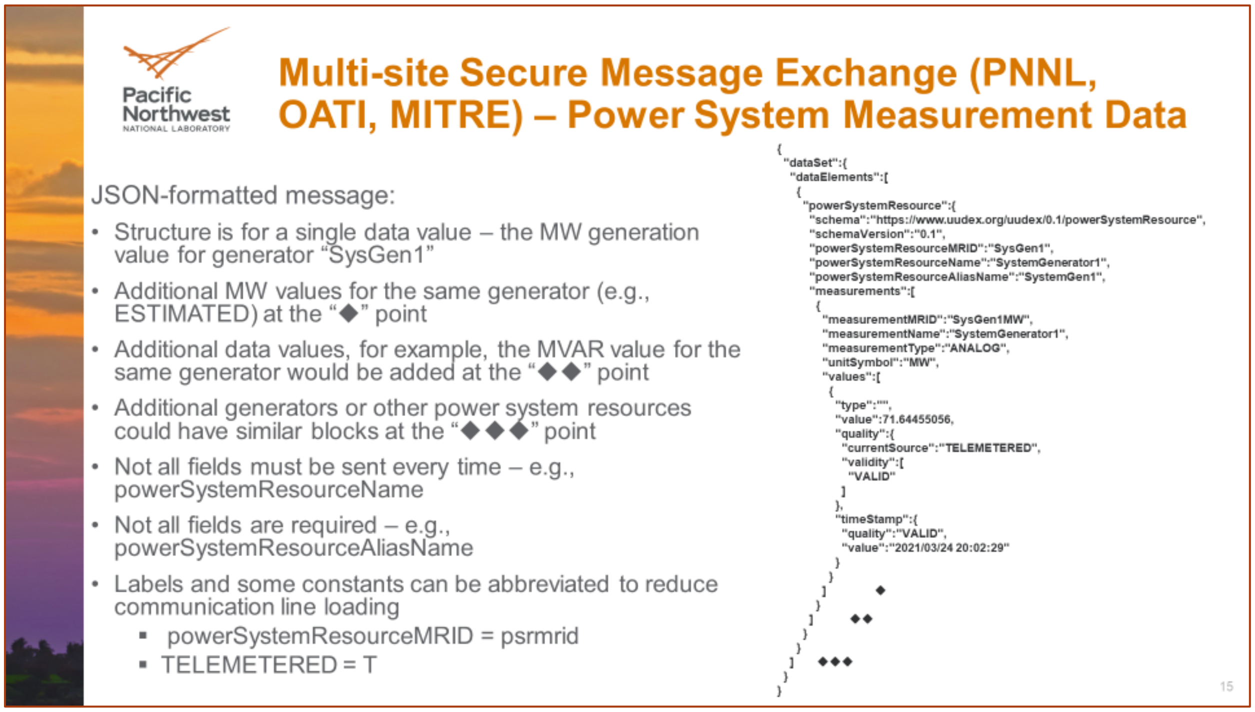
13



Multi-site Secure Message Exchange (PNNL, OATI, MITRE) – Power System Measurement Data

- Demonstrate exchange of power system measurement data between two utilities
  - PowerWorld application used to solve power flow
  - Excel spreadsheet used to simulate one-line display
- Data is formatted using JSON data structures created by UUDEX project
- Power System Measurements use IEC 61970 Common Information Model (CIM) Master Resource Identifiers (mRIDs) as universal identifiers
  - Only changed data needs to be sent
  - Data can be sent in arbitrary order
  - Demonstration sends all data for coding simplicity
- Demonstrate Subscription processing
  - Subscription is fulfilled whenever data is published to one of its Subjects
- For the demonstration, PNNL both publishes and subscribes to the data using three separate “applications”
- Demonstration shows how pre-existing subjects and subscriptions can be used to exchange data

14



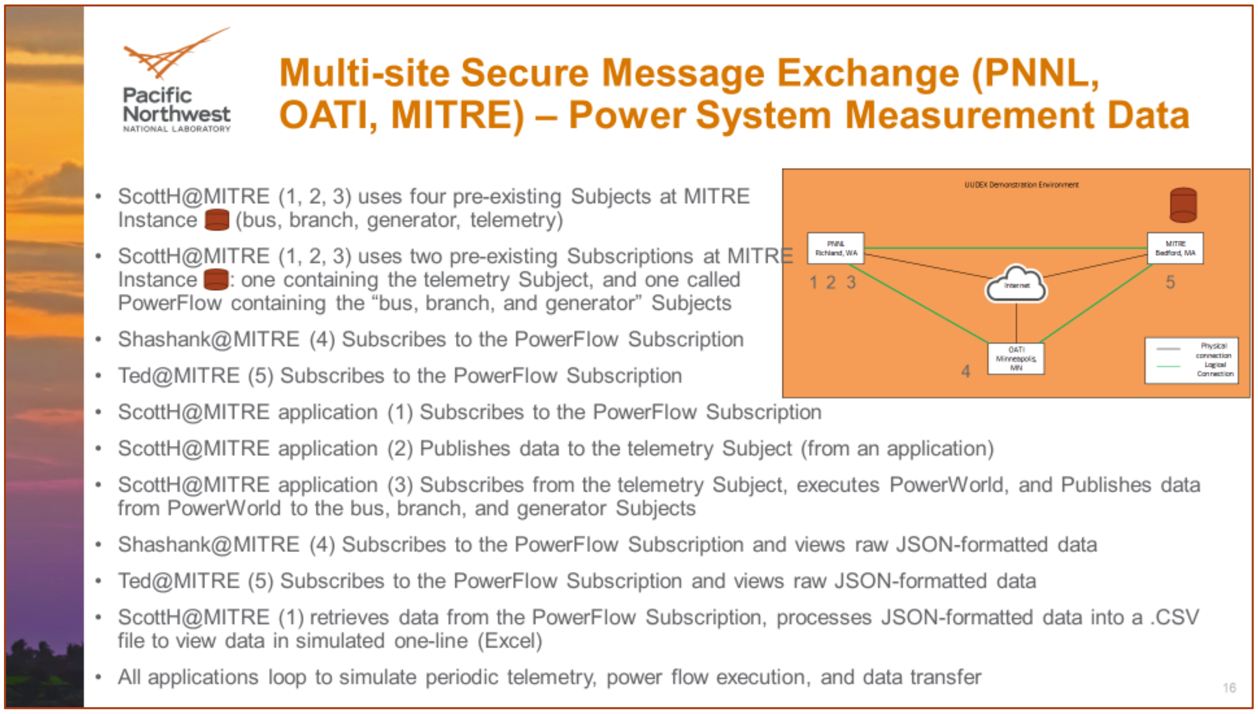
Multi-site Secure Message Exchange (PNNL, OATI, MITRE) – Power System Measurement Data

JSON-formatted message:

- Structure is for a single data value – the MW generation value for generator "SysGen1"
- Additional MW values for the same generator (e.g., ESTIMATED) at the "◆" point
- Additional data values, for example, the MVAR value for the same generator would be added at the "◆◆" point
- Additional generators or other power system resources could have similar blocks at the "◆◆◆" point
- Not all fields must be sent every time – e.g., powerSystemResourceName
- Not all fields are required – e.g., powerSystemResourceAliasName
- Labels and some constants can be abbreviated to reduce communication line loading
  - powerSystemResourceMRID = psrmrid
  - TELEMETERED = T

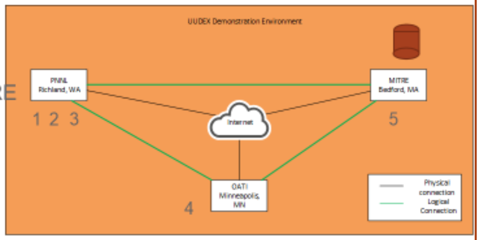
```
{
 "dataSet":{
  "dataElements":[
   {
    "powerSystemResource":{
     "schema":"https://www.uudex.org/uudex/0.1/powerSystemResource",
     "schemaVersion":"0.1",
     "powerSystemResourceMRID":"SysGen1",
     "powerSystemResourceName":"SystemGenerator1",
     "powerSystemResourceAliasName":"SystemGen1",
     "measurements":[
      {
       "measurementMRID":"SysGen1MW",
       "measurementName":"SystemGenerator1",
       "measurementType":"ANALOG",
       "unitSymbol":"MW",
       "values":[
        {
         "type":"",
         "value":71.64455056,
         "quality":{
          "currentSource":"TELEMETERED",
          "validity":[
           "VALID"
          ]
         },
         "timeStamp":{
          "quality":"VALID",
          "value":"2021/03/24 20:02:29"
         }
        }
       ]     ◆
      }
     ]     ◆◆
    }
   }
  ]   ◆◆◆
 }
}
```

15

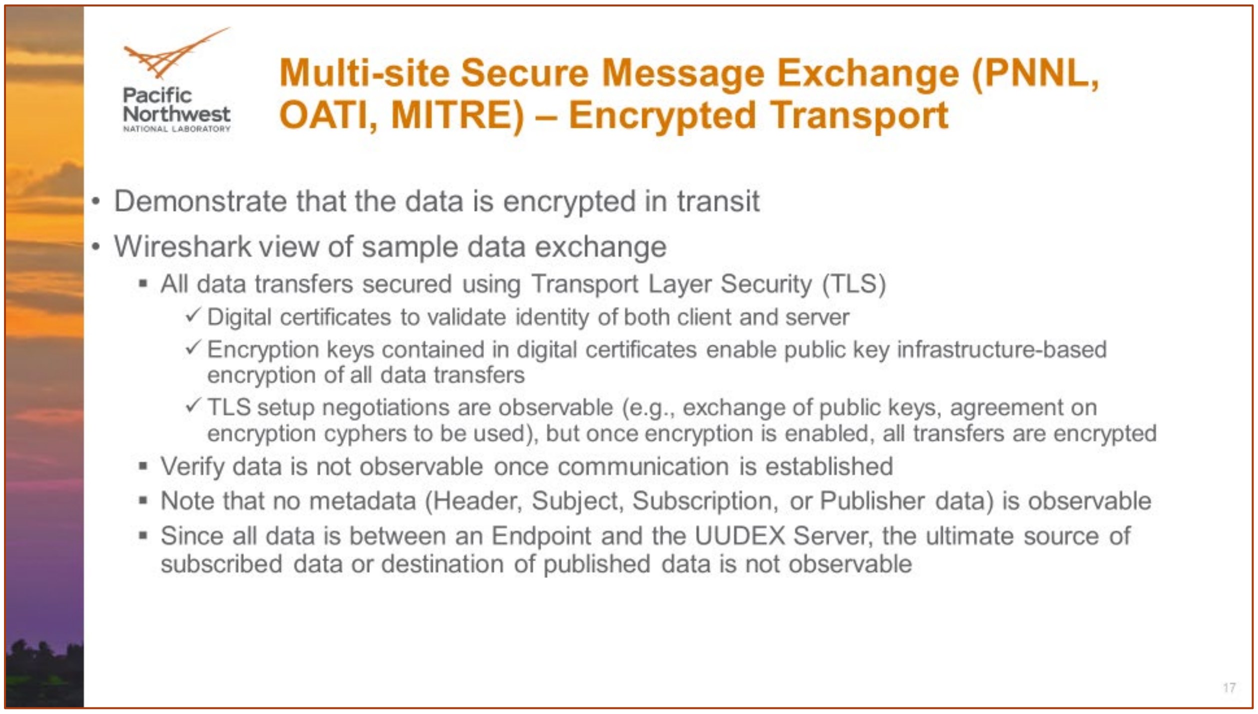


Multi-site Secure Message Exchange (PNNL, OATI, MITRE) – Power System Measurement Data

- ScottH@MITRE (1, 2, 3) uses four pre-existing Subjects at MITRE Instance (bus, branch, generator, telemetry)
- ScottH@MITRE (1, 2, 3) uses two pre-existing Subscriptions at MITRE Instance : one containing the telemetry Subject, and one called PowerFlow containing the "bus, branch, and generator" Subjects
- Shashank@MITRE (4) Subscribes to the PowerFlow Subscription
- Ted@MITRE (5) Subscribes to the PowerFlow Subscription
- ScottH@MITRE application (1) Subscribes to the PowerFlow Subscription
- ScottH@MITRE application (2) Publishes data to the telemetry Subject (from an application)
- ScottH@MITRE application (3) Subscribes from the telemetry Subject, executes PowerWorld, and Publishes data from PowerWorld to the bus, branch, and generator Subjects
- Shashank@MITRE (4) Subscribes to the PowerFlow Subscription and views raw JSON-formatted data
- Ted@MITRE (5) Subscribes to the PowerFlow Subscription and views raw JSON-formatted data
- ScottH@MITRE (1) retrieves data from the PowerFlow Subscription, processes JSON-formatted data into a .CSV file to view data in simulated one-line (Excel)
- All applications loop to simulate periodic telemetry, power flow execution, and data transfer



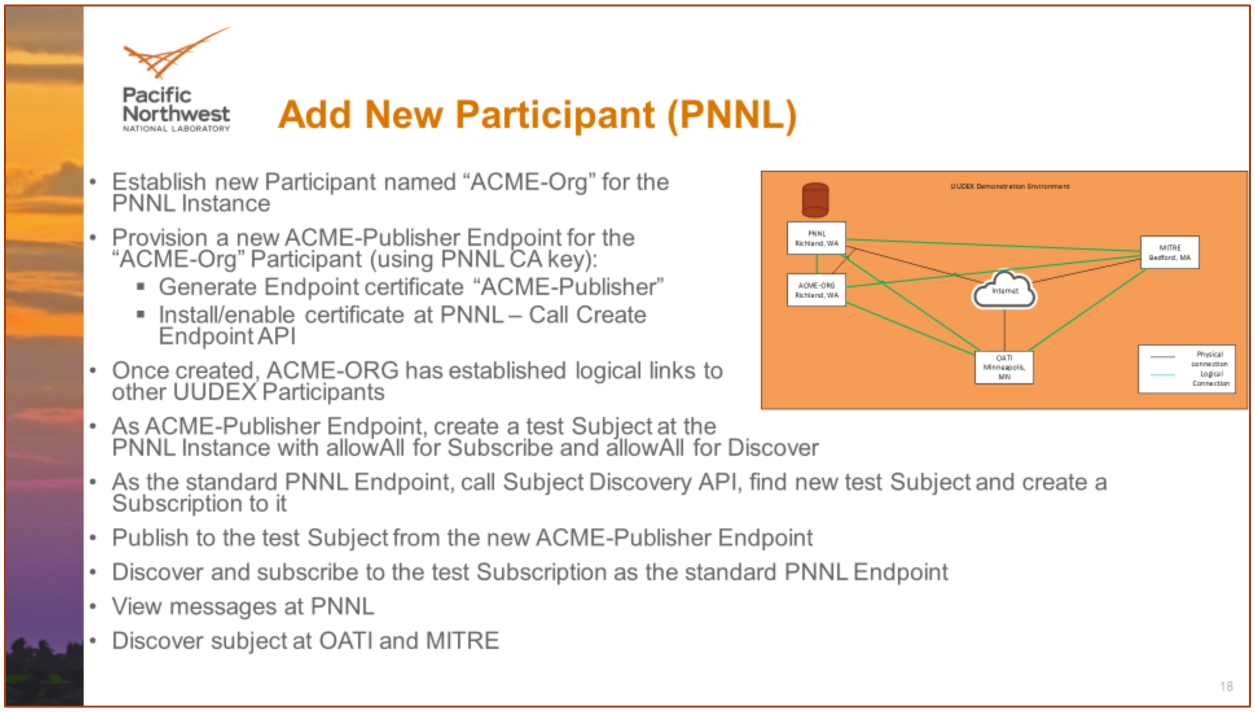
16



Multi-site Secure Message Exchange (PNNL, OATI, MITRE) – Encrypted Transport

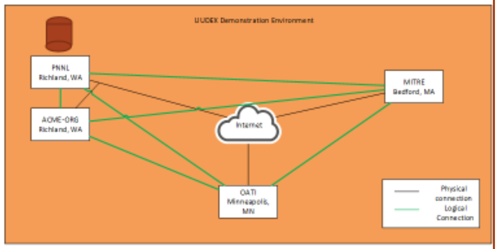
- Demonstrate that the data is encrypted in transit
- Wireshark view of sample data exchange
  - All data transfers secured using Transport Layer Security (TLS)
    - ✓ Digital certificates to validate identity of both client and server
    - ✓ Encryption keys contained in digital certificates enable public key infrastructure-based encryption of all data transfers
    - ✓ TLS setup negotiations are observable (e.g., exchange of public keys, agreement on encryption cyphers to be used), but once encryption is enabled, all transfers are encrypted
  - Verify data is not observable once communication is established
  - Note that no metadata (Header, Subject, Subscription, or Publisher data) is observable
  - Since all data is between an Endpoint and the UUDEX Server, the ultimate source of subscribed data or destination of published data is not observable

17

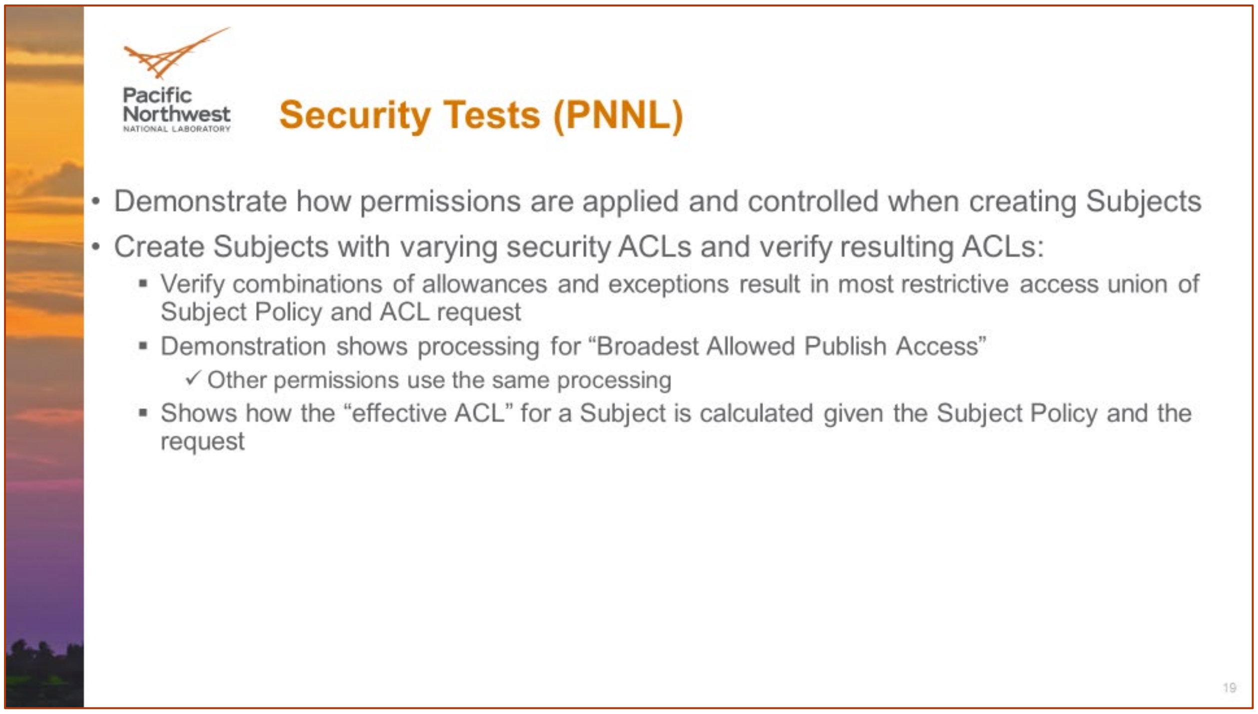


Add New Participant (PNNL)

- Establish new Participant named "ACME-Org" for the PNNL Instance
- Provision a new ACME-Publisher Endpoint for the "ACME-Org" Participant (using PNNL CA key):
  - Generate Endpoint certificate "ACME-Publisher"
  - Install/enable certificate at PNNL – Call Create Endpoint API
- Once created, ACME-ORG has established logical links to other UUDEX Participants
- As ACME-Publisher Endpoint, create a test Subject at the PNNL Instance with allowAll for Subscribe and allowAll for Discover
- As the standard PNNL Endpoint, call Subject Discovery API, find new test Subject and create a Subscription to it
- Publish to the test Subject from the new ACME-Publisher Endpoint
- Discover and subscribe to the test Subscription as the standard PNNL Endpoint
- View messages at PNNL
- Discover subject at OATI and MITRE



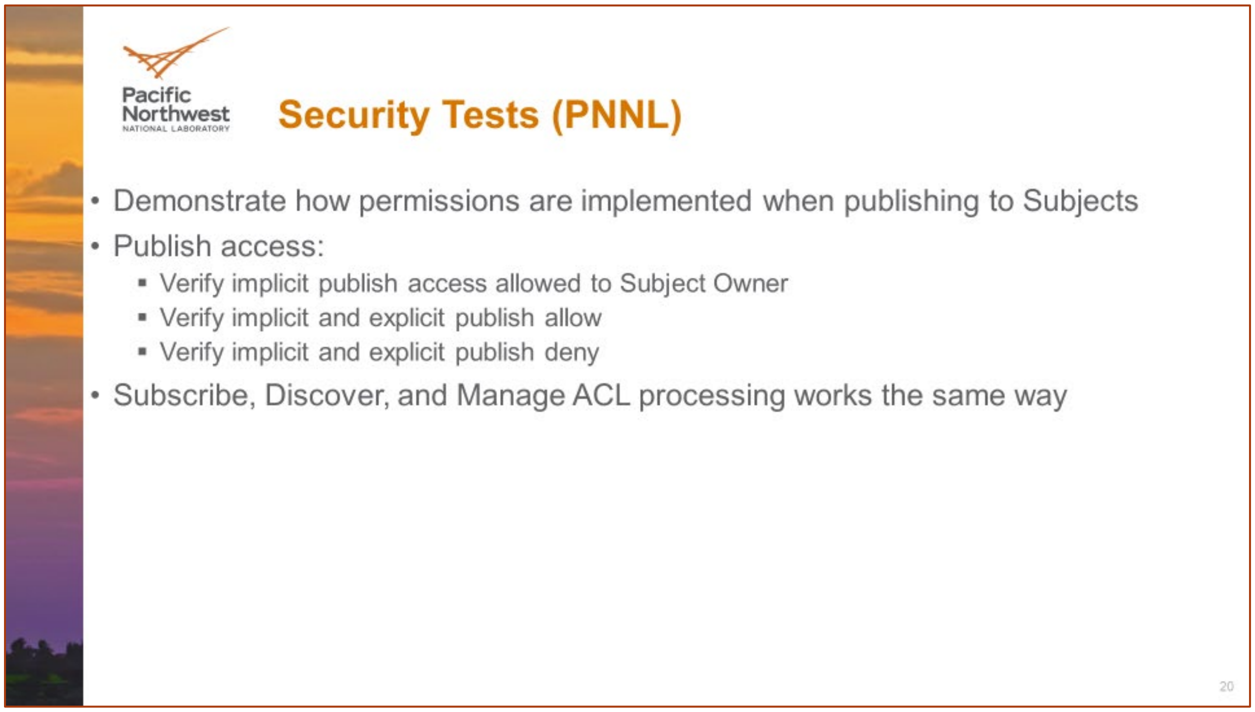
18



Security Tests (PNNL)

- Demonstrate how permissions are applied and controlled when creating Subjects
- Create Subjects with varying security ACLs and verify resulting ACLs:
  - Verify combinations of allowances and exceptions result in most restrictive access union of Subject Policy and ACL request
  - Demonstration shows processing for "Broadest Allowed Publish Access"
    - ✓ Other permissions use the same processing
  - Shows how the "effective ACL" for a Subject is calculated given the Subject Policy and the request

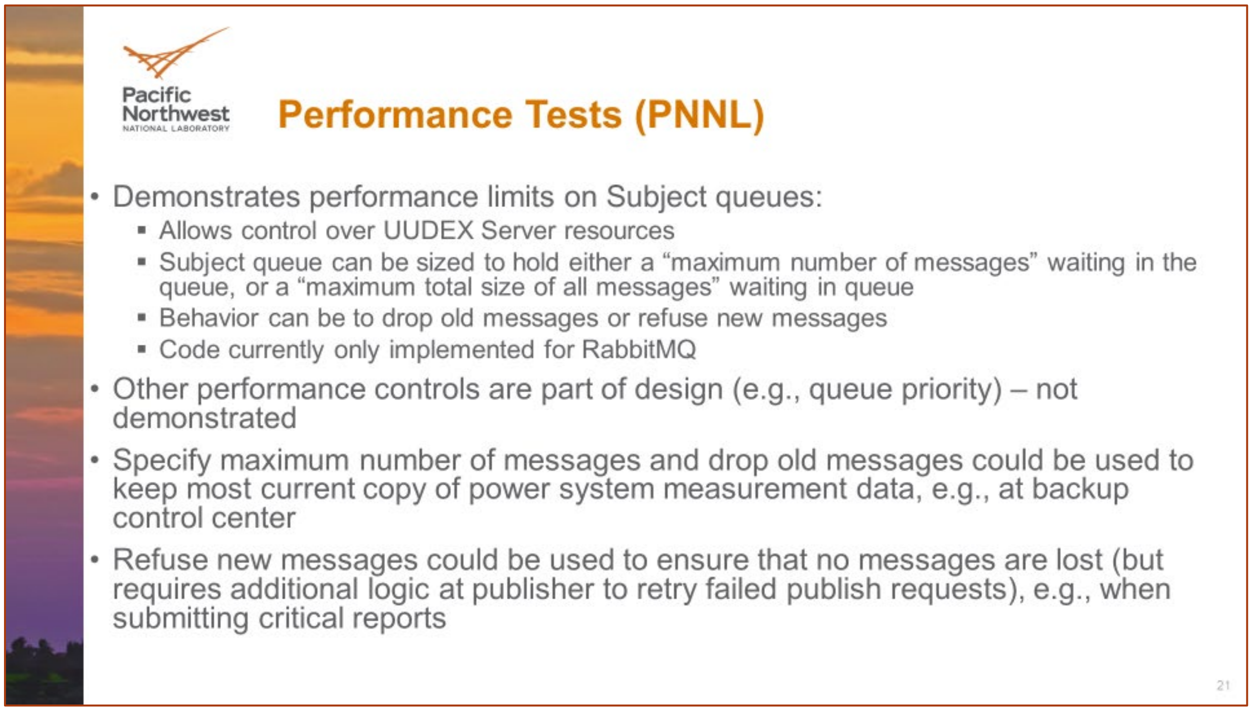
19



Security Tests (PNNL)

- Demonstrate how permissions are implemented when publishing to Subjects
- Publish access:
  - Verify implicit publish access allowed to Subject Owner
  - Verify implicit and explicit publish allow
  - Verify implicit and explicit publish deny
- Subscribe, Discover, and Manage ACL processing works the same way

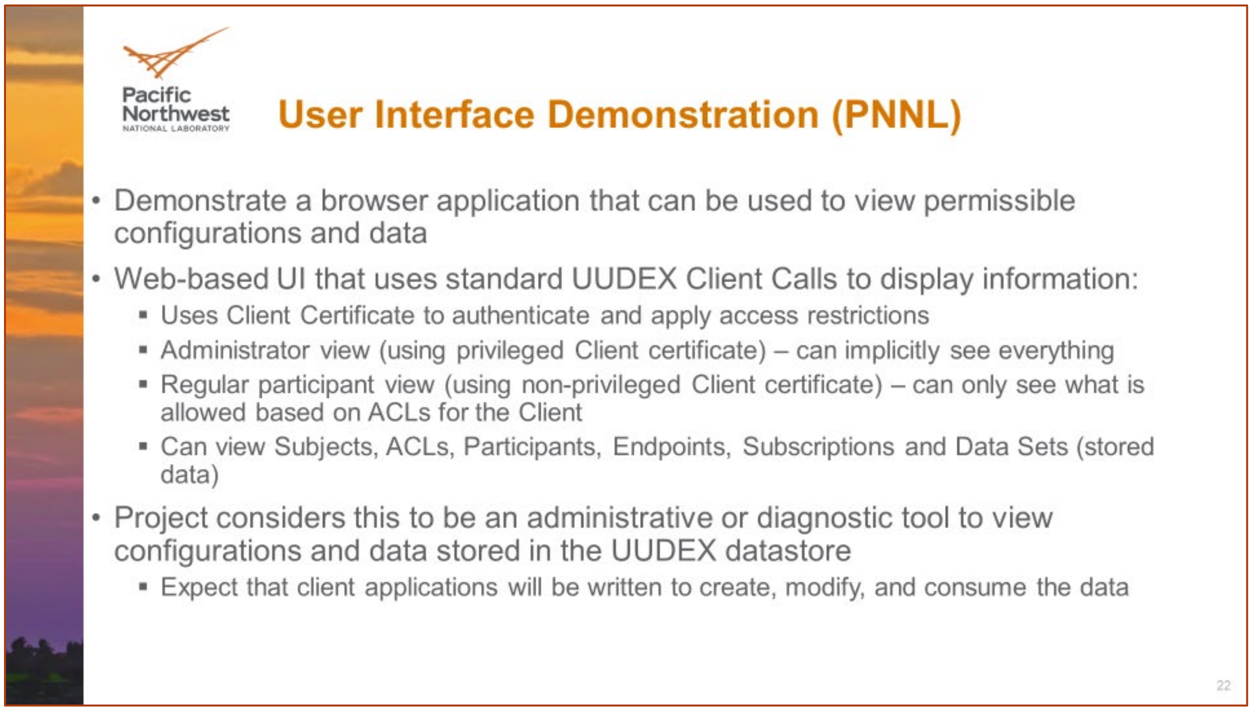
20



Performance Tests (PNNL)

- Demonstrates performance limits on Subject queues:
  - Allows control over UUDEX Server resources
  - Subject queue can be sized to hold either a "maximum number of messages" waiting in the queue, or a "maximum total size of all messages" waiting in queue
  - Behavior can be to drop old messages or refuse new messages
  - Code currently only implemented for RabbitMQ
- Other performance controls are part of design (e.g., queue priority) – not demonstrated
- Specify maximum number of messages and drop old messages could be used to keep most current copy of power system measurement data, e.g., at backup control center
- Refuse new messages could be used to ensure that no messages are lost (but requires additional logic at publisher to retry failed publish requests), e.g., when submitting critical reports

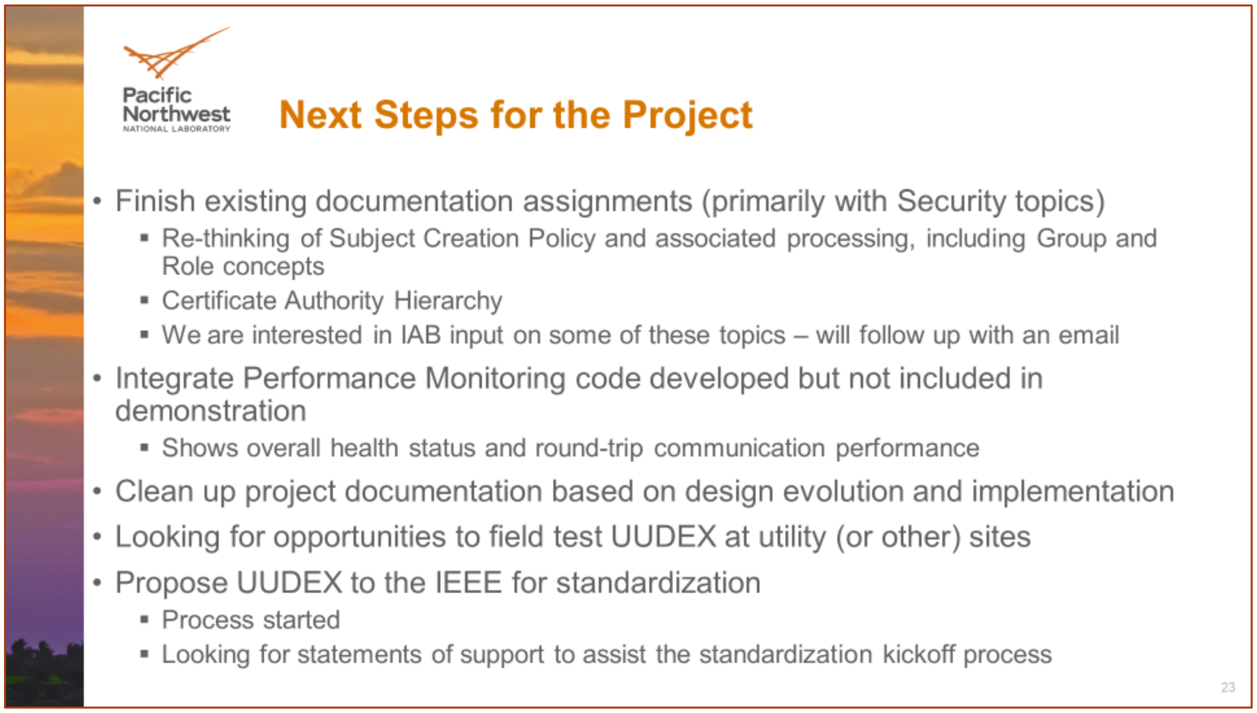
21



User Interface Demonstration (PNNL)

- Demonstrate a browser application that can be used to view permissible configurations and data
- Web-based UI that uses standard UUDEX Client Calls to display information:
  - Uses Client Certificate to authenticate and apply access restrictions
  - Administrator view (using privileged Client certificate) – can implicitly see everything
  - Regular participant view (using non-privileged Client certificate) – can only see what is allowed based on ACLs for the Client
  - Can view Subjects, ACLs, Participants, Endpoints, Subscriptions and Data Sets (stored data)
- Project considers this to be an administrative or diagnostic tool to view configurations and data stored in the UUDEX datastore
  - Expect that client applications will be written to create, modify, and consume the data

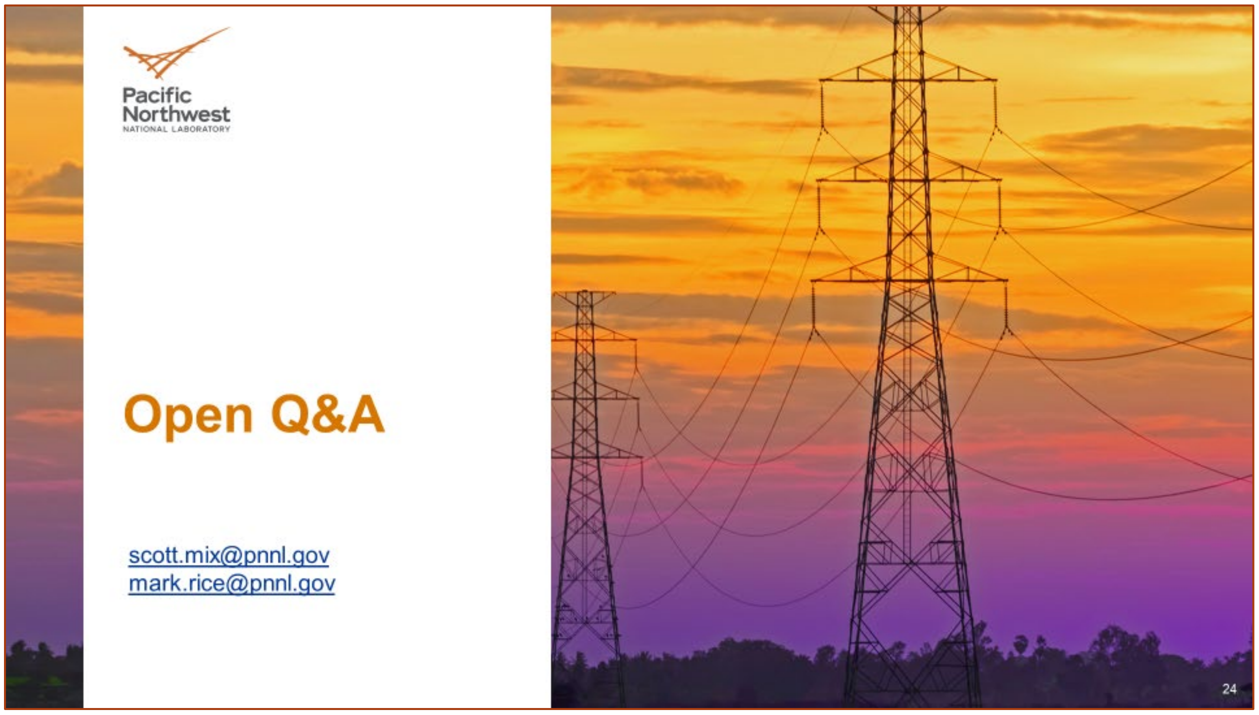
22



Next Steps for the Project

- Finish existing documentation assignments (primarily with Security topics)
  - Re-thinking of Subject Creation Policy and associated processing, including Group and Role concepts
  - Certificate Authority Hierarchy
  - We are interested in IAB input on some of these topics – will follow up with an email
- Integrate Performance Monitoring code developed but not included in demonstration
  - Shows overall health status and round-trip communication performance
- Clean up project documentation based on design evolution and implementation
- Looking for opportunities to field test UUDEX at utility (or other) sites
- Propose UUDEX to the IEEE for standardization
  - Process started
  - Looking for statements of support to assist the standardization kickoff process

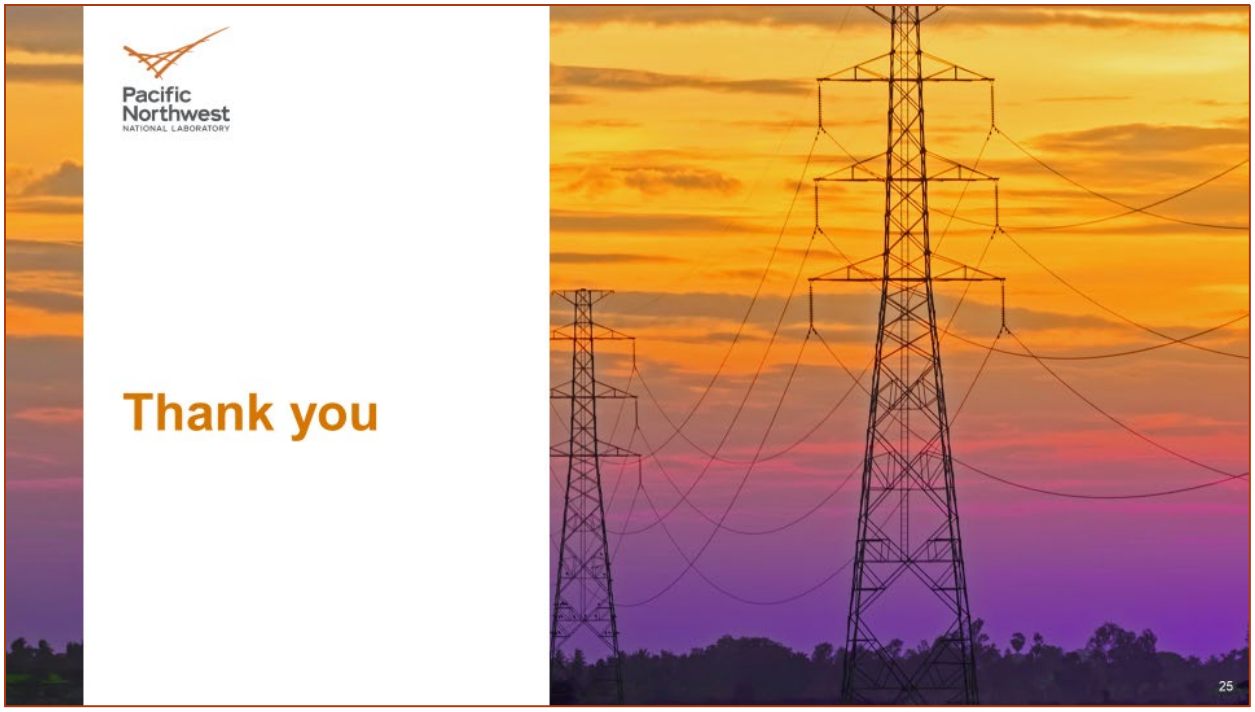
23



Open Q&A

scott.mix@pnnl.gov
mark.rice@pnnl.gov

24



Pacific Northwest NATIONAL LABORATORY

Thank you

25