

SEADTrain Workflow User Manual

1. Overview

1.1. Purpose of SEADTrain

1.2. Architecture

1.3. SEAD Workflow

1.4. RPID Testbed

1.5. PIDs

2. Demo

2.1. Get Collection

2.1.1. MQTT client

2.1.2. Sensor data type

2.2. Publishing

2.2.1. SEAD Workflow

2.2.1.1. SEAD Client

2.2.1.2. SEAD Curbee

2.2.1.3. IU SEAD Cloud

2.2.1.3.1. PIDs of ROs

2.2.1.3.2. Discovery User Interface

2.2.1.3.3. Azure Blobs

3. Analysis Hands-on

3.1. Help session

3.2. Analysis Hands-on

1. Overview

1.1. Purpose of SEADTrain

Microsoft Azure SEADTrain is to create and evaluate an environment for data analysis that allows students to interact with data. It will extend SEAD's publishing tool suite to support Azure as a destination and will use a persistent identification framework (PIDs) to reference datasets at varying granularity.

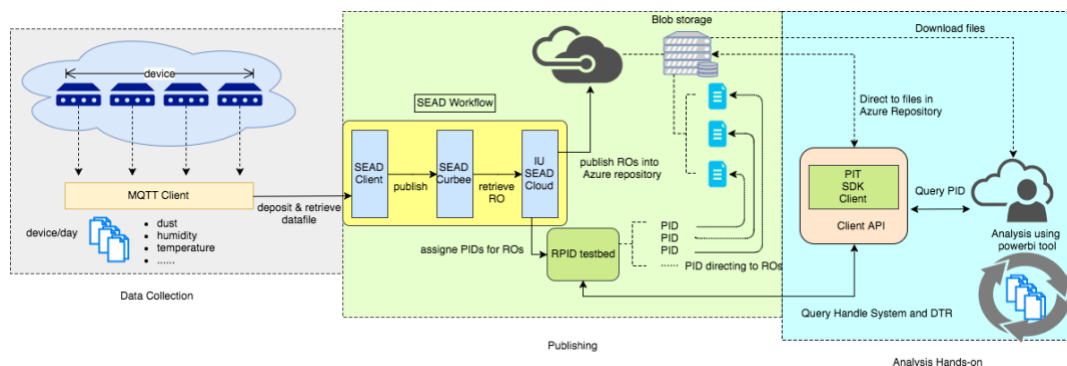
SEADTrain project publishes data from AirBox devices for the purposes of creating data science learning modules. We assign a PID to each daily feed from one device, generating daily files per device, per day from the raw readings. For further analysis, this data needs to be queried and subset for specific time ranges.

The challenge is making this PID-enabled environment user-friendly and scalable, which ensures faster resolution and secure analysis. Such user-friendly environment will also help researchers with different backgrounds to use Microsoft Azure.

1.2. Architecture

Three main components in SEADTrain:

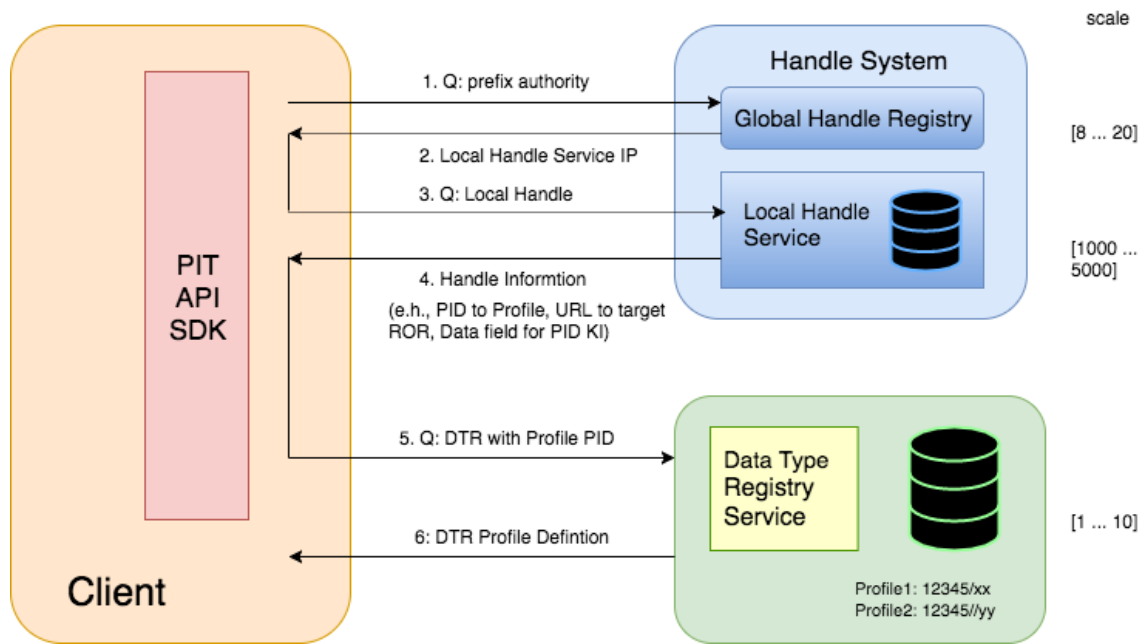
- Data collection
 - Download the raw sensor data from Airbox devices into the server machine.
- Publishing
 - Publish the raw data through SEAD Workflow, assign PIDs to each raw data file, and deposit the raw data files into Azure Blob storage
- Analysis Hands-on
 - Use some tools analyzing raw data, gain insights into data.



1.3. SEAD Workflow

SEAD Workflow is a workflow combining three components: SEAD Client, SEAD Curbee and IU SEAD Cloud. The input of workflow is the raw sensor data, converting data files into ROs (Research Object), and the output of workflow is the PIDs referencing files in Azure Blobs.

1.4. RPID Testbed



Our RPID Testbed, in progress, will

- Store PID Kernel Information (see section 1.5) in Local Handle Service
- Utilize **Data Type Registry (DTR)**, endorsed by RDA, to store the profile of PID Kernel Information
- Utilize **PIT API**, endorsed by RDA, as an SDK of tools for clients to interact with PID infrastructure

1.5. PIDs

Persistent Identifiers (PIDs) are globally unique IDs with a strong governance structure around them to ensure resolving authority. Our work uses the Handle System for PIDs.

The Handle allows for a small amount of extensible defining metadata (that we call PID kernel information). This information, defined wisely, can enable an entirely new ecosystem of data services operating at Internet speeds.

PID Kernel Information uses the **Strawman** profile (below tables) as inside metadata.

	Type of Content	Content format	Mandatory?	Explanation		Type of Content	Content Format	Mandatory?	Explanation
1	PID	Handle	YES	Global identifier for the object; external to the PID Kernel Information	1	wasDerivedFrom	IDENTIFIER	False	Transformation of an entity into another, an update of an entity resulting in a new one, or the construction of a new entity based on a pre-existing entity.
2	RDAPProfileType	Handle	YES	Handle to the Kernel Information type profile; serves as pointer to profile in DTR. Address of DTR federation expected to be global (common) knowledge.	2	specializationOf	IDENTIFIER	False	Entity is of another shares all aspects of the latter, and additionally presents more specific aspects of the same thing as the latter.
3	digitalObjectType	Handle	YES	Handle points to type defin in DTR. The type of the object (this should always be the same for this type of data, but would distinguish it from other data types). Distinguishing metadata from data objects is a client decision within a particular usage context, which may to some extent rely on the digitalObjectType value provided.	3	revisionOf	IDENTIFIER	False	A derivation for which the resulting entity is a revised version of some original.
4	digitalObjectLocation	URL	YES	Pointer to the content object location (pointer to DO)	4	primarySourceOf	IDENTIFIER	False	Used for a topic refers to something produced by some agent with direct experience and knowledge about the topic, at the time of the topic's study, without benefit from hindsight.
5	etag	Hex String	YES	Checksum of object contents	5	quotationOf	IDENTIFIER	False	Used for the repeat of (some or all of) an entity, such as text or image, by someone who may or may not be its original author.
6	lastModified	ISO Date	YES	Last time of digital object modification	6	alternateOf	IDENTIFIER	False	Entities present aspects of the same thing. These aspects may be the same or different, and the alternate entities may or may not overlap in time.
7	creationDate	ISO Date	YES	Date of digital object	7	hadMember	IDENTIFIER	False	A membership relation is defined for stating the members of a Collection.
8	version	String	YES	If tracked, a numerical version for the object	8	externalW3CPROVDoc	URL	False	A URL referring to a W3C PROV document from an external repository.

Sixteen fields are developed for PID Kernel Information, describing PID with human-readable information.

Below image is a real PID with its Kernel Information in RPID Testbed. Type is the type of inserted data, and Data field contains the values matched with the Type.

In the below, **20.5000.347/rdastrawman** is the link of the Strawman profile defined in DTR; the Data field is the matched metadata for this PID.

<div> <div>Handle resolver URL</div> <div>Handle prefix</div> <div>Unique ID</div> </div> <div> http://hdl.handle.net/11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188 </div>			
Index	Type	Timestamp	Data
Type of inserted value		Inserted value	
1	URL	2017-06-26 16:15:50z	https://iisc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/fcfb91dc-5c39-406d-ade2-ac1e470f797f.json
2	20.5000.347/rdastrawman	2017-06-26 16:15:50z	{ "digitalObjectType": "http://hdl.handle.net/20.5000.347/rdastrawman", "creationDate": "2017-06-25T00:00:00Z", "digitalObjectLocation": "https://iisc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/fcfb91dc-5c39-406d-ade2-ac1e470f797f.json", "RDAPProfileType": "http://hdl.handle.net/20.5000.347/rdastrawman", "lastModified": "2017-06-26T00:00:00Z", "etag": "667ecfad5c6f521b96b7d7f57a7786e", "PID": "http://hdl.handle.net/11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188" }

2. Demo

Usually the entire workflow is automatic. For demo purpose, we made some changes to the SEADTrain to process each step manually.

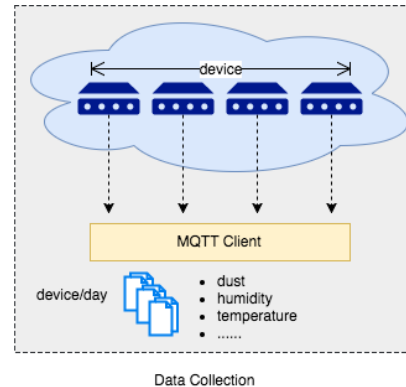
2.1. Data Collection

This section retrieves raw sensor data from the devices, chunks them by day and by device (each file has daily data for each device) as a data file.

The raw data files are saved into the server local machines without any modification on information or format.

The raw sensor data contains many values:

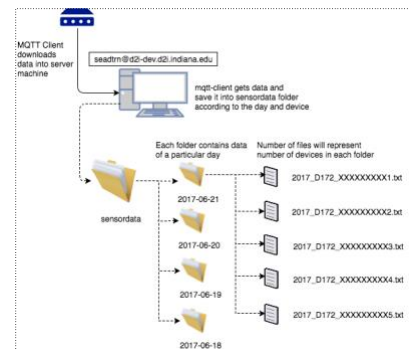
- Dust
- Humidity
- Temperature
- Barometer



2.1.1. MQTT client

The MQTT client runs in the developer server, retrieves raw sensor data from the MQTT queue, chunks them by data and by device (each file has daily data for each device), and finally saves them as a .txt file into the folder **sensordata** of developer machine.

In the **sensordata** folder of server machine, each folder contains data of a particular day. Number of files will represent number of devices in each folder.



In the developer server, we run the command lines to show the information.

```
[[seadtrn@mugo sensordata]$ du -lh
9.1M    ./2017-04-28
11M     ./2017-04-29
11M     ./2017-04-30
1.6M    ./2017-05-01
3.2M    ./2017-05-02
11M     ./2017-05-03
11M     ./2017-05-04
9.8M    ./2017-05-05
9.4M    ./2017-05-06
9.4M    ./2017-05-07
9.0M    ./2017-05-08
8.8M    ./2017-05-09
```

```
[seadtrn@mugo 2017-06-21]$ ls
2017_D172_040004cb.txt    2017_D172_9E65F90B26D7.txt    2017_D172_9E65F90C3F74.txt    2017_D172_9E65F90C537D.txt
2017_D172_040005ff.txt    2017_D172_9E65F90B2945.txt    2017_D172_9E65F90C498B.txt    2017_D172_9E65F90C5382.txt
2017_D172_9E65F90B2496.txt 2017_D172_9E65F90B2956.txt    2017_D172_9E65F90C498D.txt    2017_D172_9E65F90C53C6.txt
2017_D172_9E65F90B249D.txt 2017_D172_9E65F90C3F41.txt    2017_D172_9E65F90C5039.txt
2017_D172_9E65F90B2552.txt 2017_D172_9E65F90C3F64.txt    2017_D172_9E65F90C5378.txt
2017_D172_9E65F90B2642.txt 2017_D172_9E65F90C3F6F.txt    2017_D172_9E65F90C537C.txt
```

2.1.2. Sensor data

Four sensor data types are listed in raw sensor data files:

- d: dust sensor
 - do: G3/G5 dust sensor PM2.5
 - d1 : G3/G5 dust sensor PM10
 - d2: G3/G5 dust sensor PM1
 - d3: Panasonic SN-GCHA1 dust sensor PM2.5
- h: humidity
 - ho: DHT22 sensor
 - h1: HTS221 sensor
 - h2: SHT31 sensor
 - h3: HTU21D sensor
 - h4: BME280 sensor
 - h5: SHT25 sensor
- t: temperature
 - to: DHT22 sensor
 - t1: HTS221 sensor
 - t2: SHT31 sensor
 - t3: HTU21D sensor
 - t4: BME280 sensor
 - t5: SHT25 sensor
- b: barometer
 - bo: Grove Barometer sensor (high accuracy)
 - b1: BMP180 sensor
 - b2: BME280 sensor

In any raw sensor data file, you would see a list of messages as below image.

```
[seadtrn@mugo 2017-06-20]$ cat 2017_D171_9E65F90B2642.txt
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:00:40|device=LinkIt_Smart_7
688_Duo|tick=153314727|s_t4=30.22|s_h4=100.00|s_b2=999.82|s_d2=12|s_d0=14|s_d1=15|d_t5=36.60|d_h5=49.26|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:02:38|device=LinkIt_Smart_7
688_Duo|tick=153432750|s_t4=30.52|s_h4=100.00|s_b2=999.79|s_d2=13|s_d0=16|s_d1=16|d_t5=36.80|d_h5=48.36|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:03:37|device=LinkIt_Smart_7
688_Duo|tick=153491767|s_t4=30.41|s_h4=100.00|s_b2=999.79|s_d2=11|s_d0=14|s_d1=14|d_t5=36.84|d_h5=48.46|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:04:36|device=LinkIt_Smart_7
688_Duo|tick=153550782|s_t4=30.48|s_h4=100.00|s_b2=999.78|s_d2=14|s_d0=19|s_d1=22|d_t5=36.87|d_h5=48.77|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:07:33|device=LinkIt_Smart_7
688_Duo|tick=153727823|s_t4=30.52|s_h4=100.00|s_b2=999.77|s_d2=14|s_d0=19|s_d1=19|d_t5=36.98|d_h5=48.21|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:08:32|device=LinkIt_Smart_7
688_Duo|tick=153786851|s_t4=30.65|s_h4=100.00|s_b2=999.82|s_d2=13|s_d0=15|s_d1=17|d_t5=37.00|d_h5=47.56|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:09:31|device=LinkIt_Smart_7
688_Duo|tick=153845862|s_t4=30.62|s_h4=100.00|s_b2=999.84|s_d2=13|s_d0=16|s_d1=19|d_t5=37.02|d_h5=47.80|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:10:30|device=LinkIt_Smart_7
688_Duo|tick=153904877|s_t4=30.56|s_h4=100.00|s_b2=999.84|s_d2=15|s_d0=19|s_d1=20|d_t5=37.05|d_h5=48.45|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:12:28|device=LinkIt_Smart_7
688_Duo|tick=154022914|s_t4=30.38|s_h4=100.00|s_b2=999.88|s_d2=13|s_d0=16|s_d1=17|d_t5=37.11|d_h5=48.24|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:13:27|device=LinkIt_Smart_7
688_Duo|tick=154081929|s_t4=30.47|s_h4=100.00|s_b2=999.82|s_d2=13|s_d0=16|s_d1=20|d_t5=37.13|d_h5=47.80|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:14:26|device=LinkIt_Smart_7
688_Duo|tick=154140940|s_t4=30.72|s_h4=100.00|s_b2=999.84|s_d2=11|s_d0=15|s_d1=15|d_t5=37.16|d_h5=47.41|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:15:25|device=LinkIt_Smart_7
688_Duo|tick=154199955|s_t4=30.53|s_h4=100.00|s_b2=999.83|s_d2=12|s_d0=14|s_d1=14|d_t5=37.18|d_h5=47.17|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:17:23|device=LinkIt_Smart_7
688_Duo|tick=154317985|s_t4=30.72|s_h4=100.00|s_b2=999.85|s_d2=12|s_d0=17|s_d1=20|d_t5=37.23|d_h5=47.24|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:19:22|device=LinkIt_Smart_7
688_Duo|tick=154436016|s_t4=30.83|s_h4=100.00|s_b2=999.83|s_d2=13|s_d0=16|s_d1=19|d_t5=37.30|d_h5=47.95|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:22:19|device=LinkIt_Smart_7
688_Duo|tick=154613059|s_t4=31.06|s_h4=100.00|s_b2=999.78|s_d2=11|s_d0=13|s_d1=15|d_t5=37.40|d_h5=47.05|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:23:18|device=LinkIt_Smart_7
688_Duo|tick=154672076|s_t4=30.89|s_h4=100.00|s_b2=999.84|s_d2=13|s_d0=16|s_d1=16|d_t5=37.41|d_h5=47.33|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.8|device_id=9E65F90B2642|date=2017-06-20|time=00:27:14|device=LinkIt_Smart_7
688_Duo|tick=154908149|s_t4=30.64|s_h4=100.00|s_b2=999.84|s_d2=11|s_d0=14|s_d1=17|d_t5=37.44|d_h5=46.96|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
```

One line message contains abundant information, such like:

```
|ver_format=3|FAKE_GPS=1|app=MAPS|ver_app=5.1.5|
device_id=9E65F90B2642|date=2017-03-30|time=06:31:55|
device=LinkIt_Smart_7688_Duo|tick=704098095|
s_t4=27.94|s_h4=50.51|s_b2=1009.24|s_d2=47|s_d0=68|
s_d1=84|d_t5=36.98|d_h5=33.51|gps_lat=23.445963|
gps_lon=120.490021|gps_fix=1|gps_num=15
```

Device id: 9E65F90B2642

Date: 2017-03-30 06:31:55

Sensor:
t : temperature sensor
d : dust sensor
h : humidity sensor

2.2. Publishing

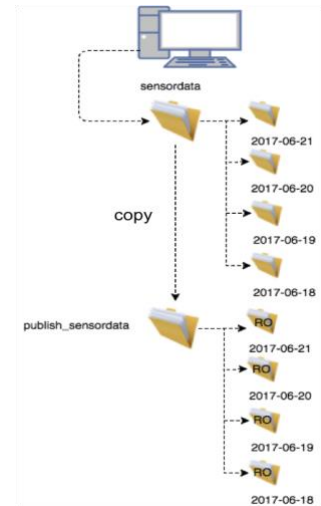
2.2.1. SEAD Workflow

The raw data are already downloaded from devices into the server machine by using MQTT client (see right image).

Before starting publishing the raw sensor data through SEAD Workflow, we are going to do a preparation.

- Copying the data folder of a particular day from **sensordata** folder into **publish_sensordata** folder that needs to be published.

```
[seadtrn@mugo mqtt-java-client]$  
[seadtrn@mugo mqtt-java-client]$ cp -r sensordata/2017-06-26 publish_sensordata/  
[seadtrn@mugo mqtt-java-client]$  
[seadtrn@mugo mqtt-java-client]$  
[seadtrn@mugo mqtt-java-client]$ cd publish_sensordata/  
[seadtrn@mugo publish_sensordata]$ ls  
2017-06-16 2017-06-17 2017-06-18 2017-06-20 2017-06-21 2017-06-23 2017-06-24 2017-06-26  
[seadtrn@mugo publish_sensordata]$
```



2.2.1.1. SEAD Client

SEAD Client checks the directory timely, for new files. If any new files are found, it will chunk and send the request to SEAD Curbee.

In the demo, we **manually** send a request with the given JSON input to URL for triggering SEAD Client to recognize the new incoming data:

<http://d2i-dev.d2i.indiana.edu:8080/sead-client/rest/streamro>.

The JSON input is below, and please change the **red part** for the new request:

```
{  
  "folder" : "/u/seadtrn/mqtt-java-client/publish_sensordata/2017-06-25/",  
  "project" : "airbox",  
  "creator" : "Kunalan",  
  "abstract" : "Airbox data for 2017-06-25",  
  "title" : "Airbox data for 2017-06-25",  
  "repository" : "iusc-azure"  
}
```

Below is the process that we send the Post request to SEAD Client.

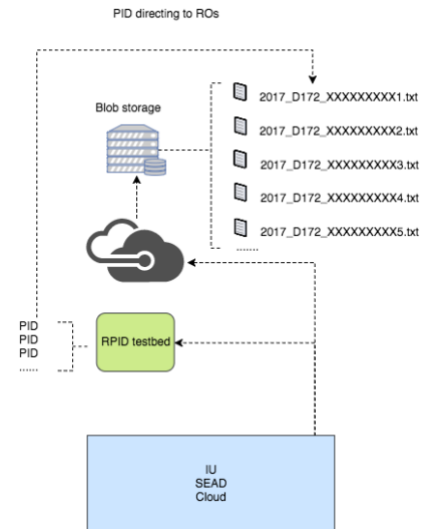
2.2.1.3. IU SEAD Cloud

IU SEAD Cloud is in charge of two events: Assign PIDs for ROs; deposit raw sensor data files into Azure Blob repository.

2.2.1.3.1. PIDs of ROs

At the end of the publishing, the IU SEAD Cloud assigns PIDs for each RO and file.

- The folder of daily device data (RO level) is assigned a **Root PID**.
- The file of raw sensor data is assigned a **Child PID**.



This is a sample Root PID:<http://hdl.handle.net/11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188>

You can use **11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188** in <http://hdl.handle.net> to view the Handle value.

Root PID

Handle.Net®			
Handle Values for: 11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188			
Index	Type	Timestamp	Data
1	URL	2017-06-26 16:15:50Z	https://iuse.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/fcfb91dc-5c39-406d-ade2-ac1e470f797f.json
2	20.5000.347/rdastrawman	2017-06-26 16:15:50Z	<pre>{ "digitalObjectType": "http://hdl.handle.net/20.5000.347/rdastrawman", "creationDate": "2017-06-25T00:00:00Z", "digitalObjectLocation": "https://iuse.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/fcfb91dc-5c39-406d-ade2-ac1e470f797f.json", "RDAKIPProfileType": "http://hdl.handle.net/20.5000.347/rdastrawman", "lastModified": "2017-06-26T00:00:00Z", "etag": "667ecfadc5c6f521b96b7d7f57a7786e", "PID": "http://hdl.handle.net/11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188" }</pre>

- The (a) part shows the unique PID in the handle value. Each object in Handle system would have a unique ID for identifying.
- Part (b) is a pair of URL type with link of the JSON metadata. This JSON file is saved in Blob repository of Azure, presenting the metadata of the folder of daily device data.
- Part (c) is our own defined DTR data type, RDA Strawman. The Strawman contains the seven non-provenance fields, describing minimal information about the RO.
- Part (d) is the matched information with the part (c).

The URL link in **Root PID** is referencing to the JSON metadata of root PID. The JSON metadata file in Azure is same as the below sample.

```

"describes": {
  "Publication Date": "07-07-2017",
  "Has Part": [
    "http://hdl.handle.net/11723/test.seadtrain.c5076e69-deff-48b7-98ec-f50135177e3c",
    "http://hdl.handle.net/11723/test.seadtrain.80f2a08d-d17a-4b5e-a80f-1d33f7fac00c",
    "http://hdl.handle.net/11723/test.seadtrain.bfbc2a65-6665-47d9-8749-f430ad072d5f",
    "http://hdl.handle.net/11723/test.seadtrain.19e26cfc-0f75-4dbd-93c8-d8592965b699",
    "http://hdl.handle.net/11723/test.seadtrain.7d65061a-b169-4343-aa2b-37a538619f5d",
    "http://hdl.handle.net/11723/test.seadtrain.3a3f04a7-e829-40fc-a2fb-25a9af70b928",
    "http://hdl.handle.net/11723/test.seadtrain.5954dc0e-da7f-45ea-9492-59686e7ffc6e",
    "http://hdl.handle.net/11723/test.seadtrain.1e7475c9-7a47-403c-81c5-d9d5343ec40f",
    "http://hdl.handle.net/11723/test.seadtrain.35cf3422-cebe-432a-9ca6-b314bdfa9c30",
    "http://hdl.handle.net/11723/test.seadtrain.ab777e6f-ef39-4b0f-a444-a4a6f39dc845",
    "http://hdl.handle.net/11723/test.seadtrain.39bbe7e-020e-4599-8538-845ab7310e6f",
    "http://hdl.handle.net/11723/test.seadtrain.c9cca651-5959-4bff-955a-97312d24a6fd",
    "http://hdl.handle.net/11723/test.seadtrain.d54ec918-1af8-4383-96f0-d8f76c4ff1b5",
    "http://hdl.handle.net/11723/test.seadtrain.3882abb1-606e-4ba0-b61f-52451a9f3ca8",
    "http://hdl.handle.net/11723/test.seadtrain.7b2eb196-e6c1-4269-8212-988f2e716821",
    "http://hdl.handle.net/11723/test.seadtrain.25d0be93-ace9-4433-98ec-a08448721da9",
    "http://hdl.handle.net/11723/test.seadtrain.8c9173b0-75a4-4aea-9395-baecb27fc509",
    "http://hdl.handle.net/11723/test.seadtrain.347f29a9-42dd-47c8-a746-28b819461433",
    "http://hdl.handle.net/11723/test.seadtrain.a75e5596-92d1-4e40-86ae-f252bf906c6f",
    "http://hdl.handle.net/11723/test.seadtrain.2bc20753-2682-481f-b424-564b0d23f015"
  ],
  "Creation Date": "2017-06-16T00:00:00Z",
  "@type": [
    "Aggregation",
    "http://cet.ncsa.uiuc.edu/2015/Dataset"
  ],
  "Creator": "Kunalan",
  "aggregates": [
    {
      "Publication Date": "07-07-2017",
      "Creation Date": "2017-06-16T00:00:00Z",
      "similarTo": "http://d2i-dev.d2i.indiana.edu:8080/fed/rest/airbox/airbox-64866018-2437-4ae4-993d-a34dfded077c/2017_D167_9E65F90B2496.txt",
      "@type": [
        "AggregatedResource",
        "http://cet.ncsa.uiuc.edu/2015/File"
      ],
      "Label": "2017_D167_9E65F90B2496.txt",
      "MimeType": "text/plain",
      "@id": "http://hdl.handle.net/11723/test.seadtrain.c5076e69-deff-48b7-98ec-f50135177e3c",
      "Last Modified": "2017-06-17T00:00:00Z",
      "Identifier": "http://hdl.handle.net/11723/test.seadtrain.c5076e69-deff-48b7-98ec-f50135177e3c",
      "Title": "2017_D167_9E65F90B2496.txt",
      "Size": 319282
    }
  ],
}

```

The **Has Part** list contains the PIDs for multiple devices of a particular day (2017-07-07). The aggregates field contains the minimal metadata about the **Child PIDs**.

Child PID

Handle.Net®		
Handle Values for: 11723/test.seadtrain.8a060692-350f-40f8-b1a7-baa9e9be60c8		
Index	Type	Timestamp
1	URL	2017-06-26 16:15:41Z
2	20.5000.347/rdastrawman	2017-06-26 16:15:41Z

The URL link in **Child PID** is referencing to a raw data file of a specified device.

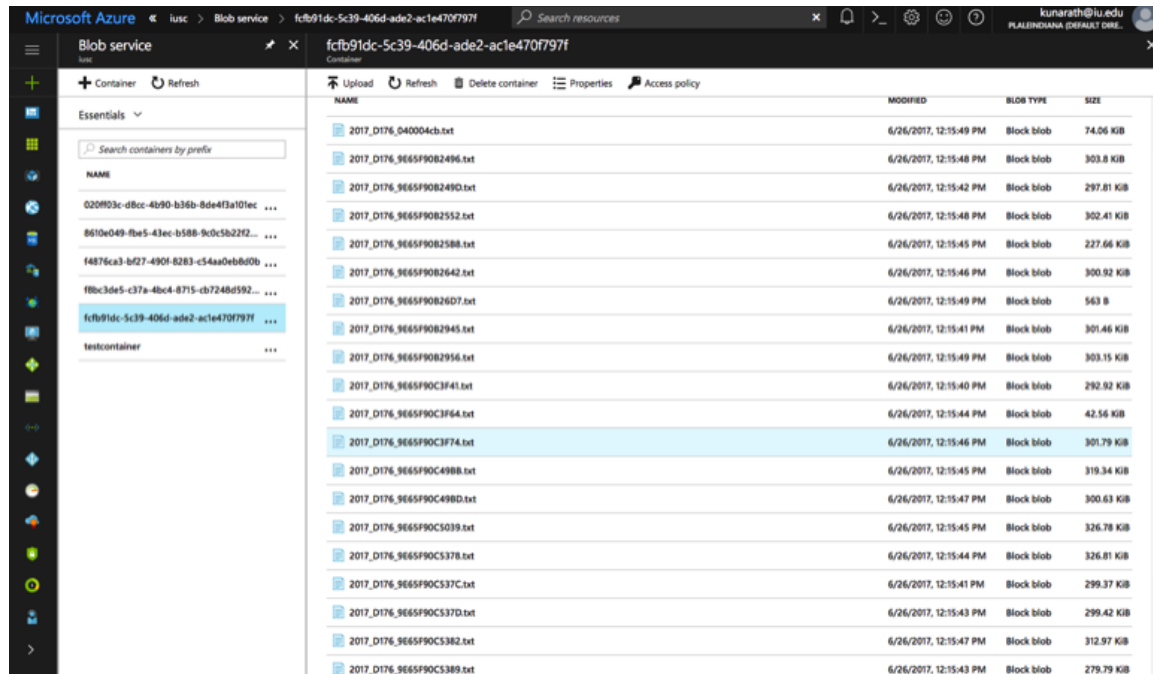
You can click on the link to download the data file directly. Or browsing the **Child PID** <http://hdl.handle.net/11723/test.seadtrain.8a060692-350f-40f8-b1a7-baa9e9be60c8>, the browser would automatically download the raw data file into your local machine without any extra process.

The PID relationship between **Root PID** and **Child PID** looks like:

In the search page, you can filter the RO list by using some key points, like publication date, creation date, title and so on. On the right list, you can see a list of results with minimal metadata including a Root PID URL.

2.2.1.3.3. Azure Blobs

Azure Blob is a stable and long-term storage for storing raw sensor data files. When the IU SEAD Cloud deposits the raw data files into Azure Blob storage, the Blob would have such list:



NAME	MODIFIED	BLOB TYPE	SIZE
2017_D176_040004cb.txt	6/26/2017, 12:15:49 PM	Block blob	74.06 KiB
2017_D176_965F90B2496.txt	6/26/2017, 12:15:48 PM	Block blob	303.8 KiB
2017_D176_965F90B249D.txt	6/26/2017, 12:15:42 PM	Block blob	297.81 KiB
2017_D176_965F90B2552.txt	6/26/2017, 12:15:48 PM	Block blob	302.41 KiB
2017_D176_965F90B2588.txt	6/26/2017, 12:15:45 PM	Block blob	227.66 KiB
2017_D176_965F90B2642.txt	6/26/2017, 12:15:46 PM	Block blob	300.92 KiB
2017_D176_965F90B26D7.txt	6/26/2017, 12:15:49 PM	Block blob	563 B
2017_D176_965F90B2945.txt	6/26/2017, 12:15:41 PM	Block blob	301.46 KiB
2017_D176_965F90B2956.txt	6/26/2017, 12:15:49 PM	Block blob	303.15 KiB
2017_D176_965F90C3F41.txt	6/26/2017, 12:15:40 PM	Block blob	292.92 KiB
2017_D176_965F90C3F64.txt	6/26/2017, 12:15:44 PM	Block blob	42.56 KiB
2017_D176_965F90C3F74.txt	6/26/2017, 12:15:46 PM	Block blob	301.79 KiB
2017_D176_965F90C4988.txt	6/26/2017, 12:15:45 PM	Block blob	319.34 KiB
2017_D176_965F90C498D.txt	6/26/2017, 12:15:47 PM	Block blob	300.63 KiB
2017_D176_965F90C5039.txt	6/26/2017, 12:15:45 PM	Block blob	326.78 KiB
2017_D176_965F90C5378.txt	6/26/2017, 12:15:44 PM	Block blob	326.81 KiB
2017_D176_965F90C537C.txt	6/26/2017, 12:15:41 PM	Block blob	299.37 KiB
2017_D176_965F90C537D.txt	6/26/2017, 12:15:43 PM	Block blob	299.42 KiB
2017_D176_965F90C5382.txt	6/26/2017, 12:15:47 PM	Block blob	312.97 KiB
2017_D176_965F90C5389.txt	6/26/2017, 12:15:43 PM	Block blob	279.79 KiB

3. Analysis Hands-on

The virtual machines are provided through a partnership between the Big Data Regional Innovation Hubs and Microsoft. The Data To Insight Center was awarded Azure credits through the Midwest Big Data Hub for SEADTrain. PID research and education.

The machines come pre-loaded with R, Python & Anaconda, and Power BI. You are able to download whatever software you wish to use. You will have access to the virtual machine for two weeks.

3.1 Help Section

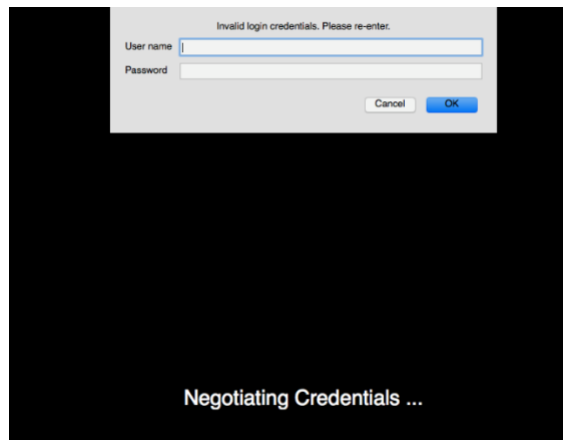
Accessing your Azure Windows VM

Receive a username, password, and link to the remote desktop protocol file from us;

- Open the link, download to your laptop, open the file and click Connect . (**Mac users will need to download Microsoft Remote Desktop in App store**)

Note: Windows OS Users: In the credentials login you need to change the username by clicking on 'more choices'--> 'Use a different account' and enter the provided username


Click Yes on the Security Certificate Confirmation screen




Mac view

Using IU SEAD CLOUD AZURE SEARCH interface:

1. To find PIDs or search the metadata associated with a PID go to:
<http://d2i-dev.d2i.indiana.edu:8081/iusc-azure-search/search.html>
 In the Title search box enter "Airbox data for year-month-day" as shown below.



INDIANA UNIVERSITY



[Learn about SEAD and its publishing services](#)

Search By:

Search Across All Fields
 search across all fields
 Ex: Airbox

Creator Name
 type creator name
 Ex: Leslie

Publication Date Range
 start date - end date
 Ex: 01/04/2016 - 04/04/2016 [mm/dd/yyyy]

Title
 Airbox data for 2017-06-21
 Ex: Airbox data for 2017-06-25

Clear All
Search

IU SEAD Cloud

Discover Research Objects Published in Azure

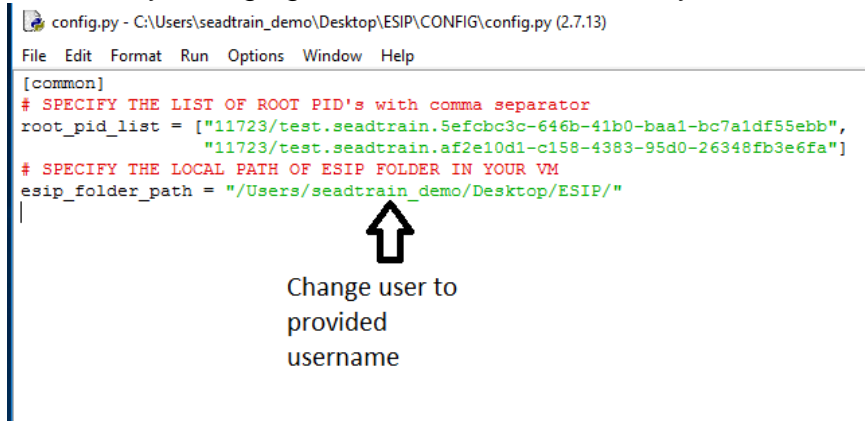
Prev 1 Next

Title	Airbox data for 2017-06-21
Creator	Kunalan
Publication Date	Jul 5, 2017 12:10:07 PM
PID	http://hdl.handle.net/11723/test.seadtrain.5ef21060-c223-4b21-a5bc-8bef348221ff
Abstract	Airbox data for 2017-06-21

2. If you have a PID and want to know what day the data was collected copy and paste it into the “Search Across All Fields” box to view the metadata.
3. Clicking on the PID link will open the json file that contains all of the child PIDs for each Airbox devices’ data for that day as well as the metadata. Feel free to copy a Child PID and put in the browser to see the raw data for one device.
4. Let’s grab the most recent PID and add it to ESIP/CONFIG/config.py on the VM. In the Publication Date Range box enter in today’s date for the start date and tomorrow’s date for the end date. Copy the PID and return to the ESIP folder to paste it within the **root_pid_list** field

Compiling the Data for Analysis

1. In the ESIP folder, open up the Config File, enter the path to your ESIP folder, only changing the username is necessary.



```
config.py - C:\Users\seadtrain_demo\Desktop\ESIP\CONFIG\config.py (2.7.13)
File Edit Format Run Options Window Help

[common]
# SPECIFY THE LIST OF ROOT PID's with comma separator
root_pid_list = ["11723/test.seadtrain.5efcbc3c-646b-41b0-baa1-bc7a1df55ebb",
                 "11723/test.seadtrain.af2e10d1-c158-4383-95d0-26348fb3e6fa"]
# SPECIFY THE LOCAL PATH OF ESIP FOLDER IN YOUR VM
esip_folder_path = "/Users/seadtrain_demo/Desktop/ESIP/"
```

↑
Change user to
provided
username

2. Save the file and close Idle.
3. Double click on pid_resolver.py to execute the program
 - Note the creation of two new folders within ESIP: PID-RESOLVER-FILES-OUTPUT and PID-RESOLVER-METADATA-OUTPUT

Cleaning Up the Files

1. Double Click on csv-convert.py to execute the script and create csv files from the text files.
2. Double Click on csv-merge.py to combine all of the separate device files within the daily RO files into one csv.
3. If you want to analyze more than one day’s data double click on the final-csv-merge.py to create ‘final.csv’ for all of the ROs that you specified.

The Data

Some devices transmit data fields that others do not, data may be missing for some devices.

Fields	Measurement & unit	Data type
PID	URI handle	string
device_id	12 character	string
date	year-month-day	date
time	hour:min:sec	time
device	name	String: LinkIt_Smart_7688_Duo
s_t4	temperature : Celsius	float: %.2f
s_h4	relative humidity : %	float: %.2f
s_b2	Barometer : [millibars]	float: %.6f
s_d2	dust sensor PM1: [ug/cm ³]	integer: 2 sig figs
s_d0	dust sensor PM2.5: [ug/cm ³]	integer: 2 sig figs
s_d1	dust sensor PM10 : [ug/cm ³]	integer: 2 sig figs
d_t5	device temperature	float: % .2f
d_h5	device humidity	float: %.2f
gps_lat	latitude	float : %.6f
gps_lon	longitude	float : %.6f
gps_fix	= 1	Integer
gps_num	# of satellites in gps fix = 15	Integer

3.2 Analysis Hands-on

In the VM we created a folder, **ESIP**, with the code necessary for retrieving and compiling the data. Instructions for preliminary data analysis are provided below. Have fun using the Power BI visualization tools.

USING POWER BI

Note: all of the visualizations and analysis in the training materials are using PIDs stored in ESIP/CONFIG/config.py. That file has been updated since the creation of these materials, but the methods are the same.

Power BI is an easy-to-use Microsoft tool, readily available in Azure, useful for preliminary data exploration and visualization. The Power BI Service for an individual's or organization's website will automatically update visuals and tables on the website as data is accrued.

Open Power BI Desktop



Importing Data

Click “Get Data” :

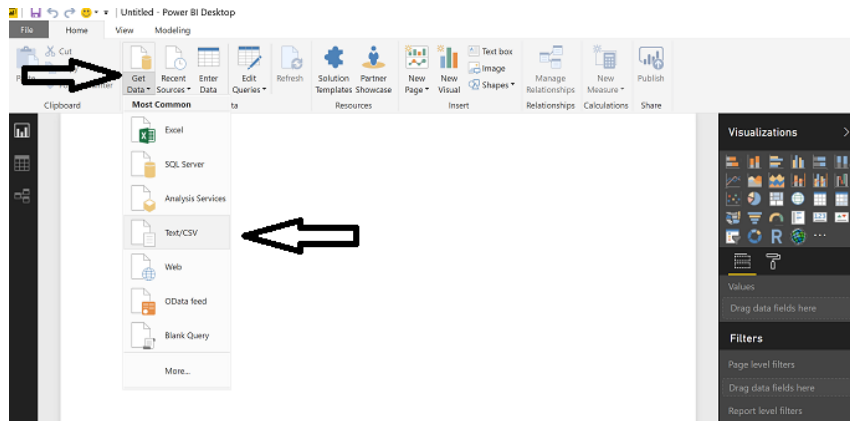


Figure 1

Select txt/csv -> connect

Select: Desktop -> ESIP -> ANALYSIS -> final.csv -> open

Check that “Column#” is NOT across the top of the table. See Figure 2 below and instructions for Editing if it is.

Otherwise Click “Load”

If Editing is needed:

Instead of clicking “Load” click “Edit”. If you already clicked “Load” you can still click “ Edit Queries” in the ribbon.

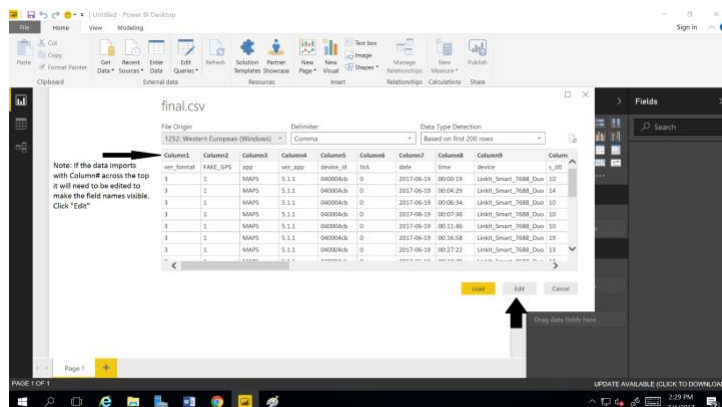


Figure 2

In the editing window click on the pull down menu next to the small table icon in the upper left corner. Select “Use First Row as Headers”. See Figure 3.

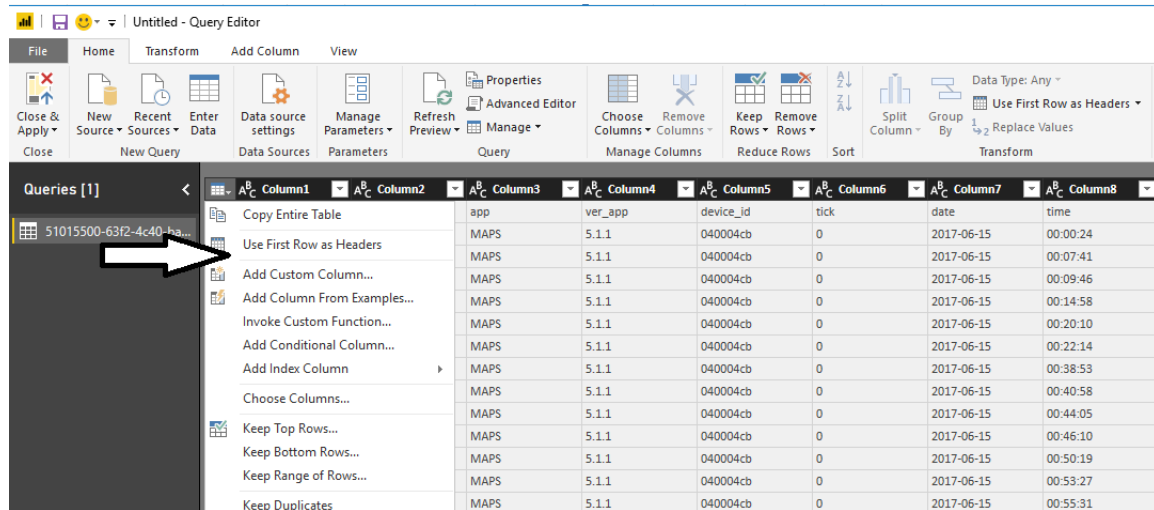


Figure 3

When the table imports the Column# as headers all of the fields are recognized as text data types. In order to manipulate numerical fields the data types for all necessary columns need to be changed to decimal or whole number. Change the s_d0, s_h4, and s_t4 to decimal. Change date to date, and time to time.

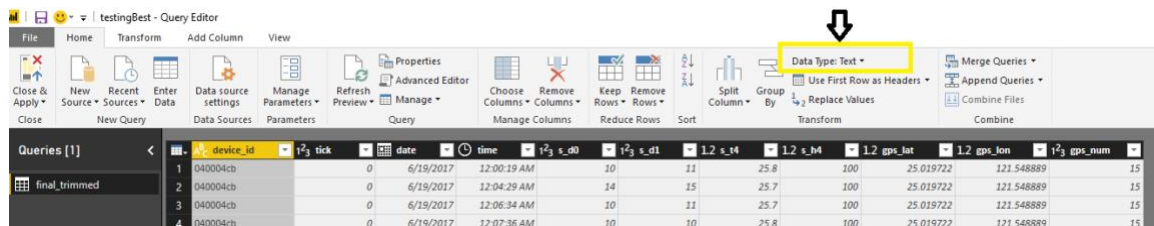


Figure 4

Making PIDs Publicly Available with Power BI

If you opted to use Power BI Service, you can export PIDs to a website by creating a table that will continuously update as new PIDs are added.

In the visualization pane click on the table icon.

Select the desired data fields that convey the necessary data.

The screenshot displays a data visualization tool interface. On the left, a data table is visible with columns: PID, device_id, Year, Month, and Day. The table contains multiple rows of data, including URLs and numerical values. On the right, a 'Visualizations' panel is active, showing a line graph icon selected. Below the graph icon, the 'Values' section lists fields: PID, device_id, date, Year, Month, and Day. The 'Filters' section includes 'Visual level filters' (date - Day(All), date - Month(All), date - Year(All), device_id(All), PID(All)) and 'Page level filters' (Drag data fields here). The 'Fields' panel on the far right shows a search bar and a list of fields: final_stripped, app, date, device, device_id, FAKE_GPS, gps_fix, gps_lat, gps_lon, gps_num, s_d0, s_d1, s_h4, s_t4, tick, time, ver_app, and ver_format.

PRELIMINARY ANALYSIS

Particulate Matter at 2.5 microns is the unit of interest. An obvious visualization is to view PM2.5 over time.

In the Visualization panel click on the Line Graph

Select the fields of interest: time, s_d0, date

They may pop up in the appropriate categories or may need to be dragged so that time is the Axis variable, date is the Legend, and s_d0 is in the Values.

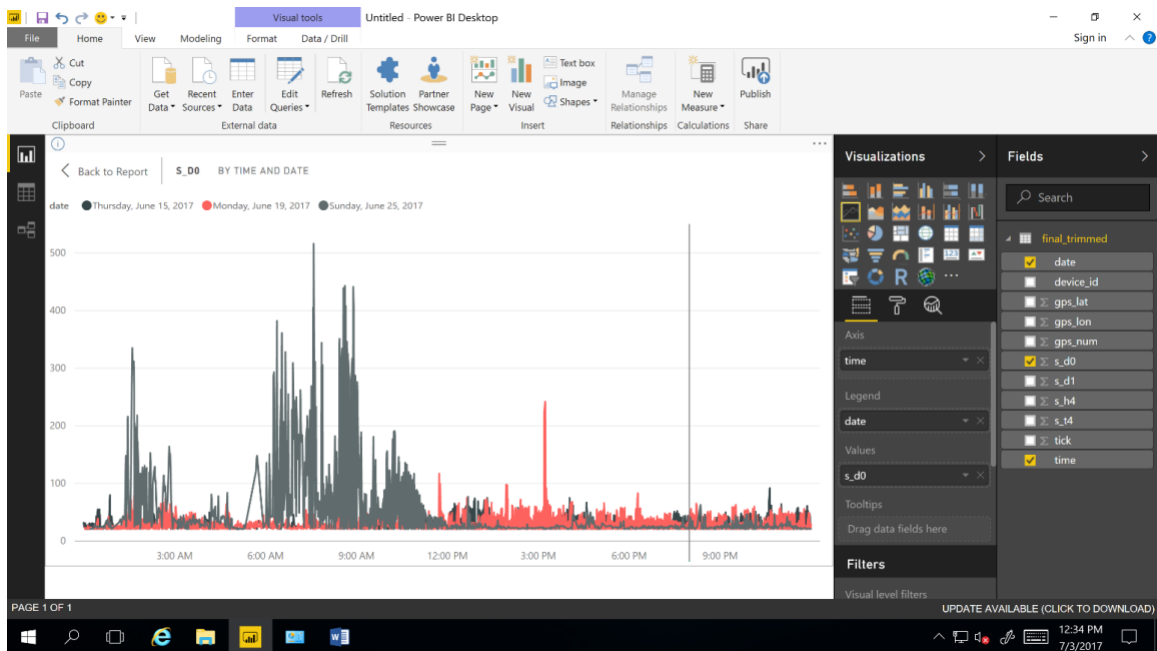


Figure 5

Let's apply a filter to get rid of the extraneous days where a measurement was taken right at midnight. Under Filters, click the drop down arrow for s_d0. Under "Show items when the value: " select "is greater than" and type in 20 directly underneath.

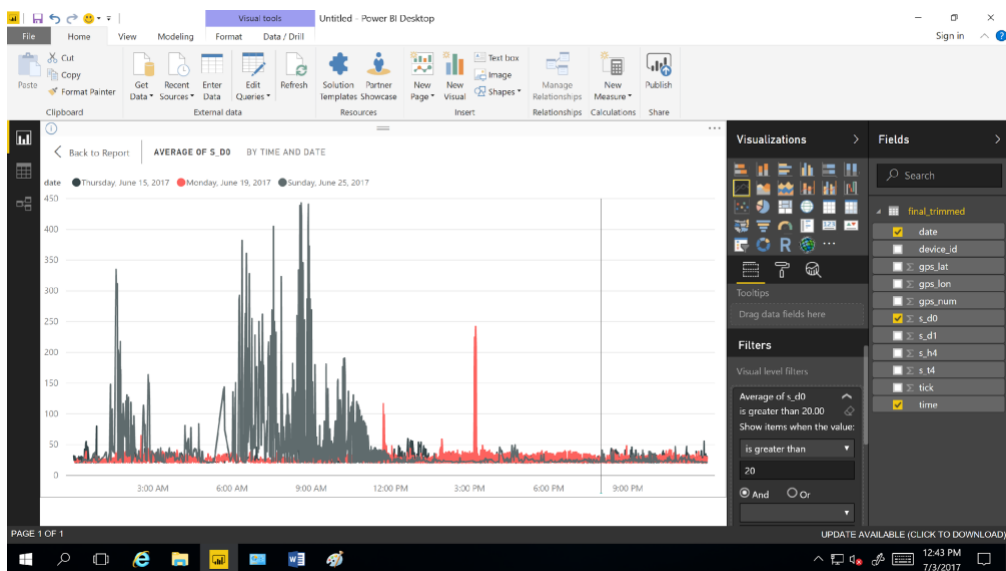


Figure 6

You can interact with the visualization by clicking through the legend to emphasize certain days. This visualization shows Sunday as the peak day??

Scatter Plot Visualization: Relative Humidity vs PM2.5

Click “<Back to Report” if you expanded the view of line graph.

Click outside of the line graph to create a new chart in the dashboard.

Let’s explore the relationship between Relative Humidity and PM2.5. This relationship changes with the size and source of the particulate matter.

Click on the “Scatter Chart” icon under Visualizations.

Under “Fields” click in this order: device_id for the Legend, s_h4 for the X axis, s_d0 for the PM2.5. Drag the date field to the Play Axis. Use s_d0 for the Size.

Power BI automatically sums over all of the values for both Relative Humidity and PM2.5 over every time measurement for each day. Feel free to examine the built-in options for data presentation.

See Figure 8 for variable settings and filters. Dates to select are Thursday June 15, 2017, Monday June 19, 2017, and Sunday June 25, 2017

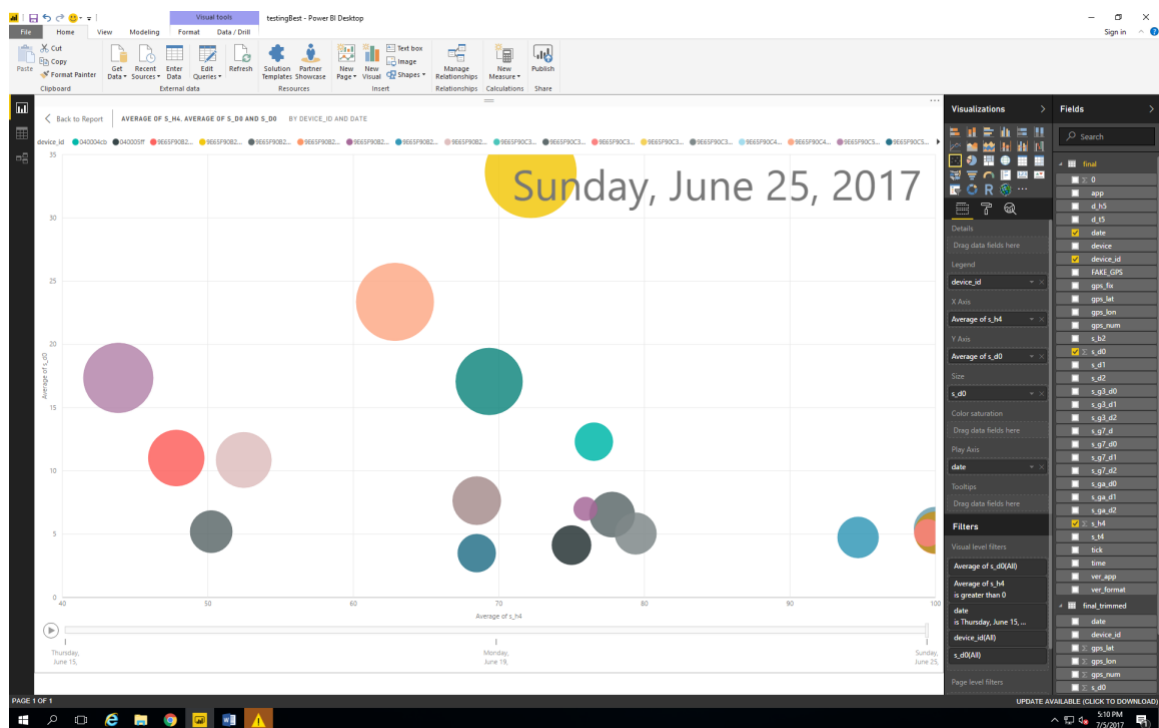


Figure 7

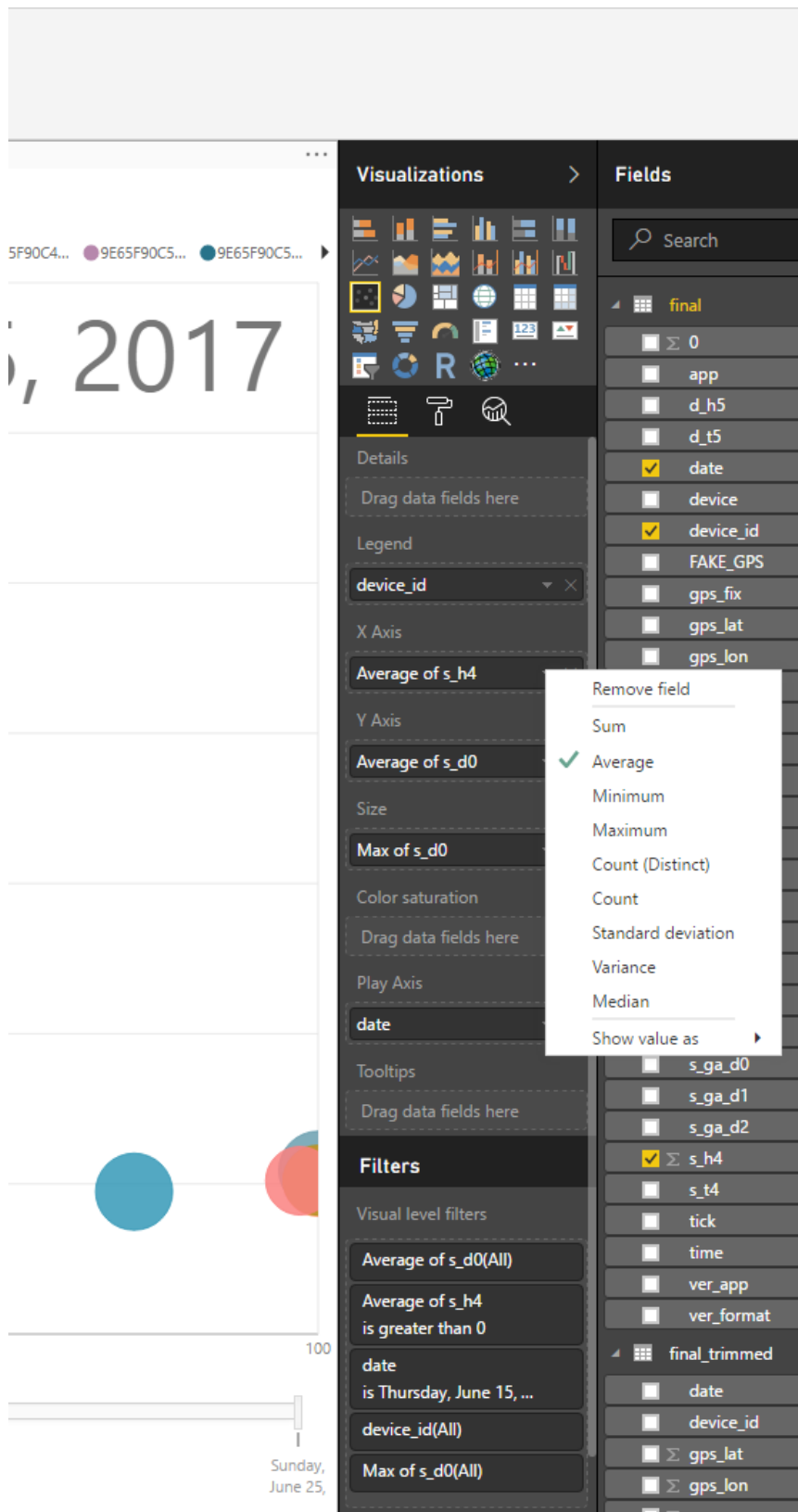


Figure 8

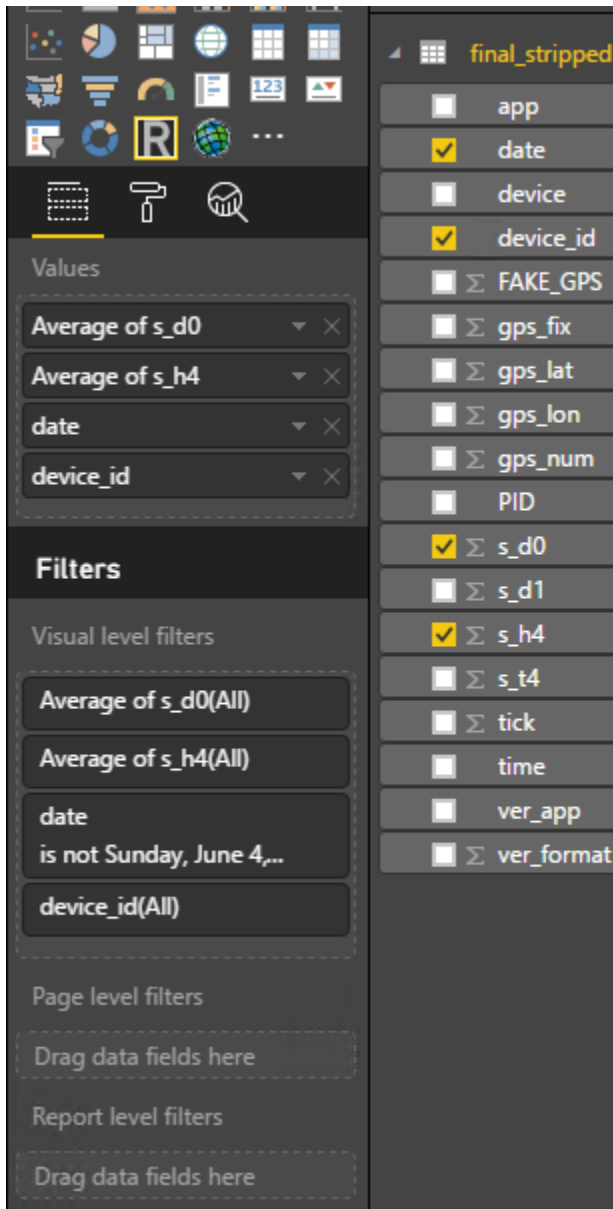
R for Statistical Analysis

Power BI does not provide statistical analysis tools except through R (or ArcGIS if you have access).

We can make a more thorough analysis of the correlation between humidity and PM2.5 given the data.

Select R in visualization pane.

Select desired data fields, they will be loaded as 'dataset' into R.



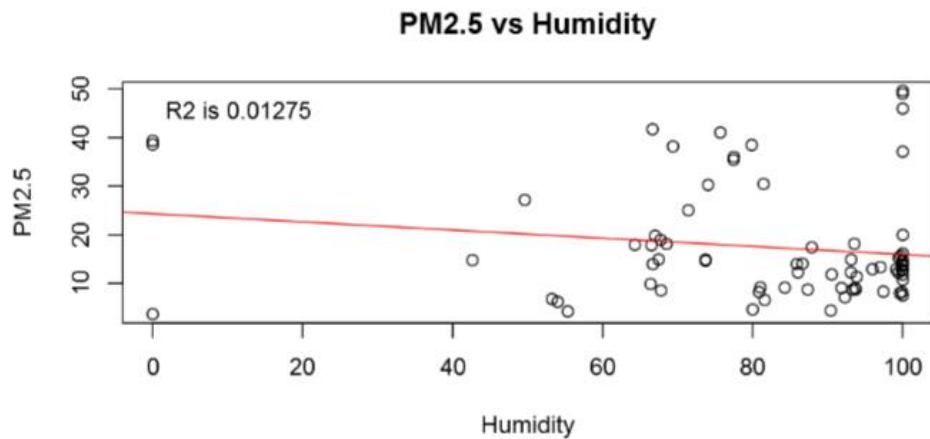
Use the R script editor to create desired plots and analysis.

<http://hdl.handle.net/11723/test.seadtrain.5138a568-8dee-41d7-b9c>
<http://hdl.handle.net/11723/test.seadtrain.572ad5fb-6cc3-45f5-a1a2>

R script editor

⚠ Duplicate rows were removed from the data.

```
# Remove duplicated rows
# dataset <- unique(dataset)
pm = dataset$s_d0
humidity = dataset$s_h4
plot(humidity, pm, xlab = "Humidity", ylab = "PM2.5", main = "PM2.5 vs Humidity")
abline(fit <- lm(pm ~ humidity), col = 'red')
legend("topleft", bty = 'n', legend = paste("R2 is", format(summary(fit)$adj.r.squared, digits = 4)))
```



Analysis Conclusion

All of the devices we are currently working with are relatively in the same geographic location and our data samples only cover a few days. We don't have a large enough sample to work with at this time. It is unlikely that the variation in PM2.5 between these devices is explained by weather conditions.

Note:

Only for Linux users

If anyone have linux machine then you need to do the below steps configure azure windows VM into your machine.

Remote Desktop Client: Remmina, <https://www.remmina.org/wp/>

Steps to install:

```
sudo snap install remmina
```

```
sudo apt-add-repository ppa:remmina-ppa-team/remmina-next
```

```
sudo apt-get update
```

```
sudo apt-get install remmina remmina-plugin-rdp libfreerdp-  
plugins-standard
```

How to use:

```
remmina -c ~/Desktop/SEADTrainDemo3.rdp
```

Useful Resources:

<https://github.com/FreeRDP/Remmina/wiki>

<http://tipsonubuntu.com/2017/01/24/install-remmina-1-2-0-ubuntu-16-04/>

<https://askubuntu.com/questions/780291/remmina-not-launching-16-04>