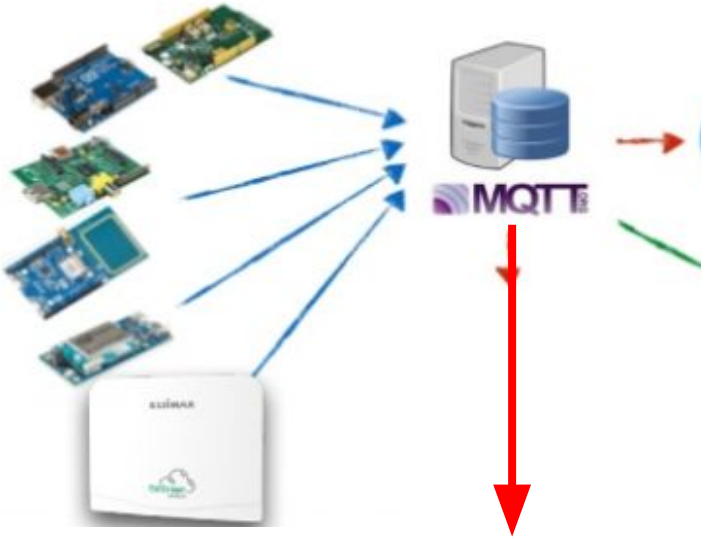
DATA TO INSIGHT CENTER

PERVASIVE TECHNOLOGY INSTITUTE

Publishing environmental sensor data: PID'ifying the Research Object

Presenters: Yu Luo & Kunalan Ratharanjan

Indiana University Bloomington, USA



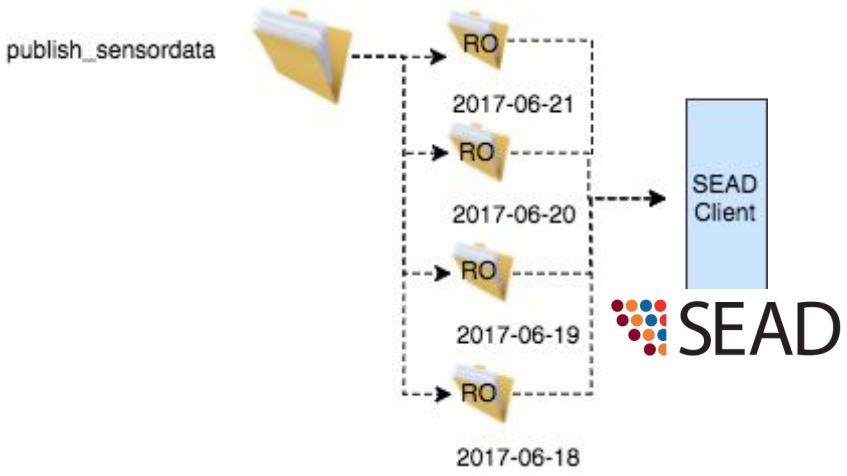
Publish device/day data

Create ORE and Submit Publication Request → Validate Research Object Metadata → Update and Persist ORE → Persist Research Object → Mint PIDs for all Objects → Publish Research Object

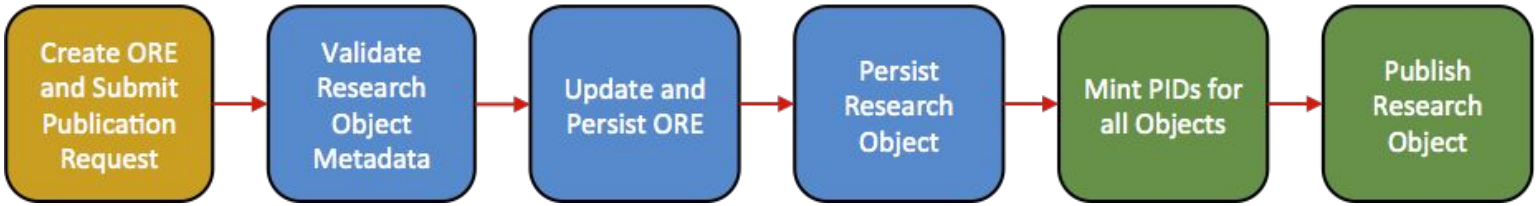
SEAD

Microsoft Azure

Step 1: group data arriving by MQTT into device/day



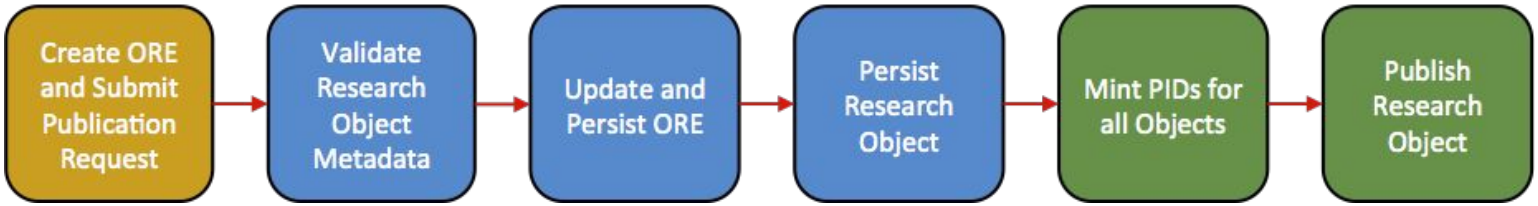
- A SEAD client checks a directory regularly for new files.
- Packages one day-one device sensor data as Research Object (RO)
- Creates ORE for RO
- Submit for publishing



Step 2: RO Validation

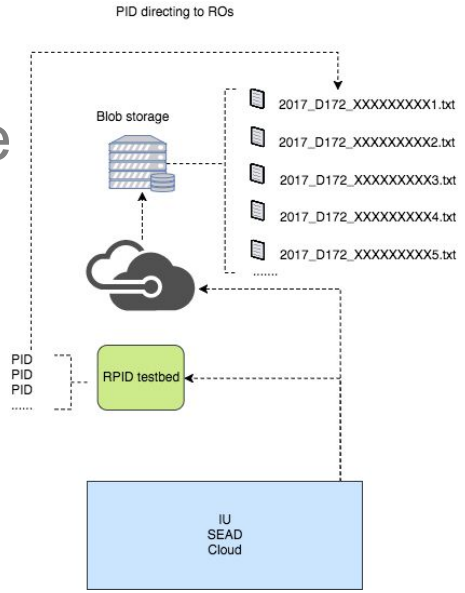
- SEAD's Curbee is a lightweight publishing workflow.
- Validates RO; RO is in JSON format, including metadata information about RO.
  - Identifier, Creation date, Publication date
  - Title, Label, Creator, Abstract
  - Size
  - File type

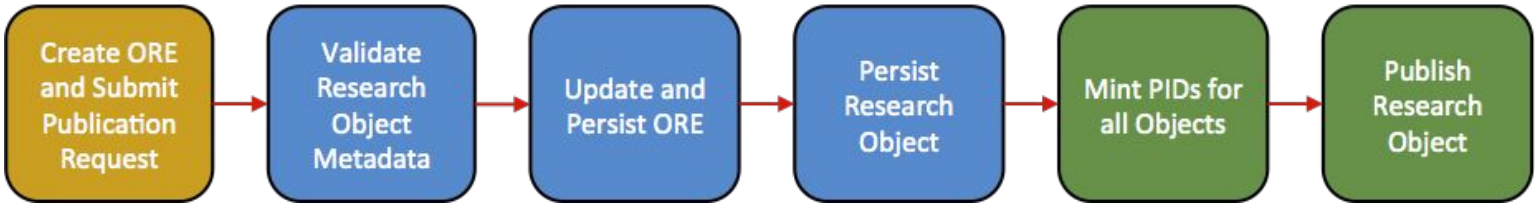
- Persists information abou ROs publshed by SEAD



Step 3: Publish to Azure

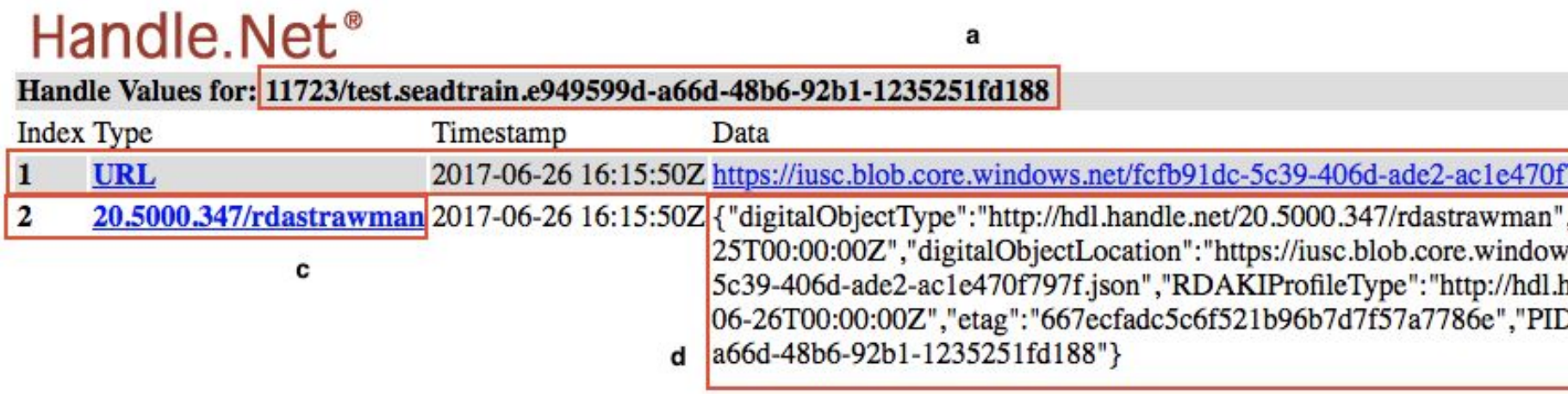
- Publishing is in two steps:
  - Execute SEAD strategy for assignment of PIDs within the RO
    - Utilize RPID Testbed to obtain handle
  - Deposit raw data into Azure Blob
    - Stable and long-term repository for storing data.





SEAD strategy for assigning PIDs within an RO**

First the RO is assign a PID that we call the **Root PID**


Handle.Net®

a

Handle Values for: 11723/test.seadtrain.e949599d-a66d-48b6-92b1-1235251fd188

Index	Type	Timestamp	Data
1	URL	2017-06-26 16:15:50Z	https://iusc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f
2	20.5000.347/rdastrawman	2017-06-26 16:15:50Z	{"digitalObjectType":"http://hdl.handle.net/20.5000.347/rdastrawman", 25T00:00:00Z","digitalObjectLocation":"https://iusc.blob.core.window 5c39-406d-ade2-ac1e470f797f.json","RDAKIProfileType":"http://hdl.h 06-26T00:00:00Z","etag":"667ecfadc5c6f521b96b7d7f57a7786e","PID a66d-48b6-92b1-1235251fd188"}

c

d

- Admin information from Handle.Net shows kernel information for the top level RO PID
- The URL link in Root PID is referencing to the JSON metadata of root PID

SEAD

SEAD strategy for assigning PIDs within an RO

The internal files are assigned **Child PIDs**.

Handle.Net®

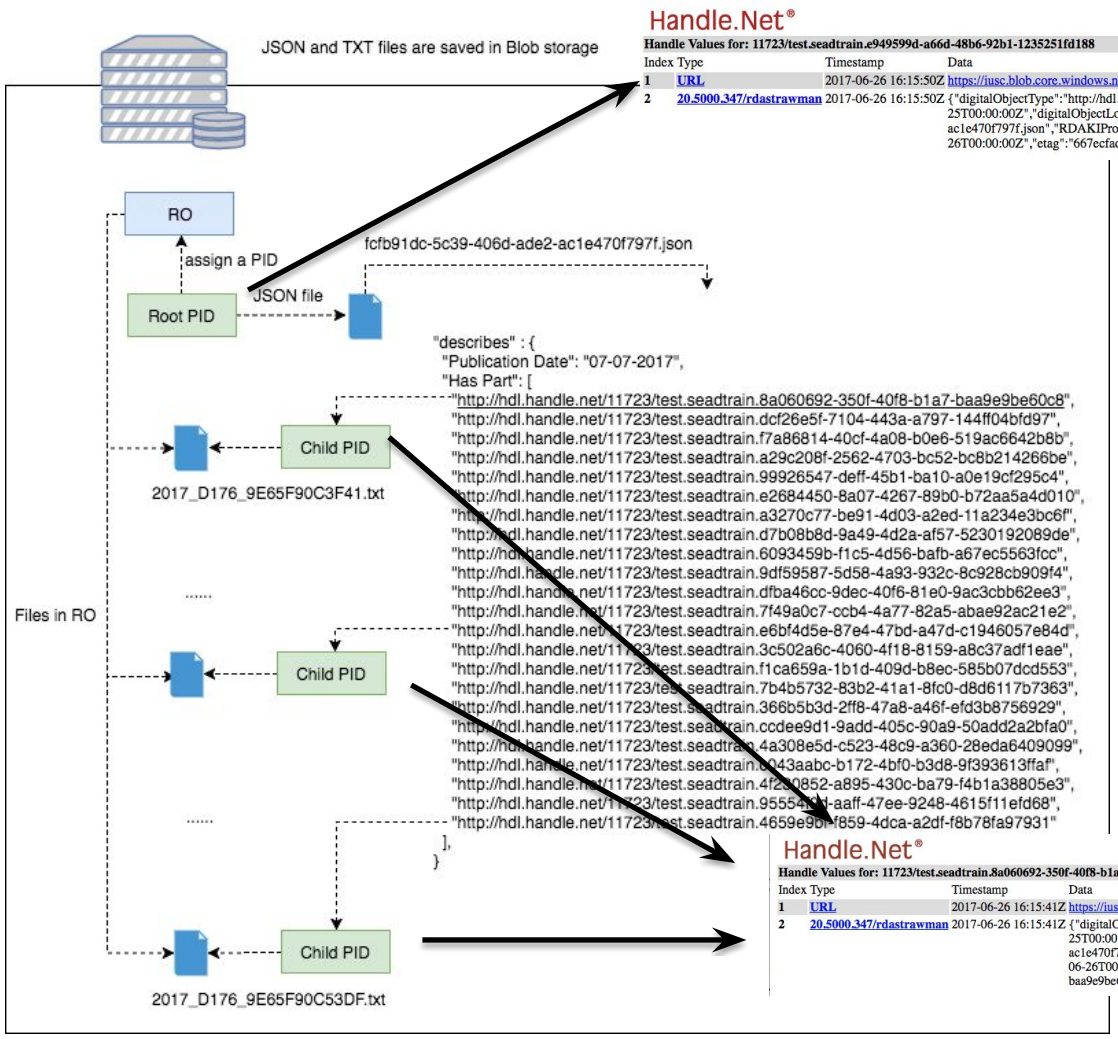
Handle Values for: 11723/test.seadtrain.8a060692-350f-40f8-b1a7-baa9e9be60c8

Index	Type	Timestamp	Data
1	URL	2017-06-26 16:15:41Z	https://iusc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/2017_D1
2	20.5000.347/rdastrawman	2017-06-26 16:15:41Z	{"digitalObjectType":"http://hdl.handle.net/20.5000.347/rdastrawman","creationDate 25T00:00:00Z" "digitalObjectLocation":"https://iusc.blob.core.windows.net/fcfb91dc ac1e470f797f/2017_D176_9E65F90C3F41.txt","RDAKIProfileType":"http://hdl.han 06-26T00:00:00Z","etag":"ba019ba61cb41d8ff2232c40d80fc5cb","PID":"http://hdl.h baa9e9be60c8"}

- The URL link in Child PID is referencing to the raw data file of one device
- Child PIDs are stored to a single ORE map file that is associated with the parent (RO)
- The parent maintains rich metadata about the RO; the children have only PID Kernel Information about them

SEAD

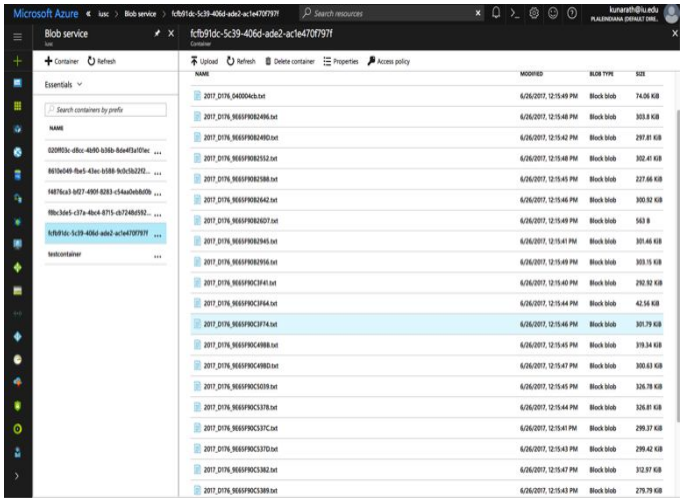
PIDs Relationship



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://iusc.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://iusc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/fcfb91dc-5c39-406d-ade2-ac1e470f797f.json","RDAKIProfileType":"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"}



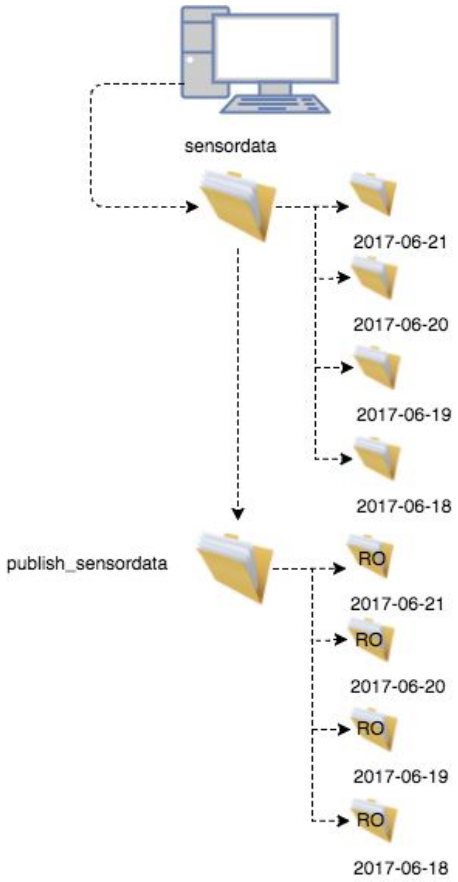
Handle.Net®

Handle Values for: 11723/test.seadtrain.8a060692-350f-40f8-b1a7-baa9e9be60c8

Index	Type	Timestamp	Data
1	URL	2017-06-26 16:15:41Z	https://iusc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/2017_D176_9E65F90C3F41.txt
2	20.5000.347/rdastrawman	2017-06-26 16:15:41Z	{"digitalObjectType":"http://hdl.handle.net/20.5000.347/rdastrawman","creationDate":"2017-06-25T00:00:00Z","digitalObjectLocation":"https://iusc.blob.core.windows.net/fcfb91dc-5c39-406d-ade2-ac1e470f797f/2017_D176_9E65F90C3F41.txt","RDAKIProfileType":"http://hdl.handle.net/20.5000.347/rdastrawman","lastModified":"2017-06-26T00:00:00Z","etag":"ba019ba61cb41d8ff2232c40d80fc5cb","PID":"http://hdl.handle.net/11723/test.seadtrain.8a060692-350f-40f8-b1a7-baa9e9be60c8"}

AirBox Ingest Demo

- Publishing through SEAD is an automatic process
- For demo purposes, we show how each step works by stepping manually through the process



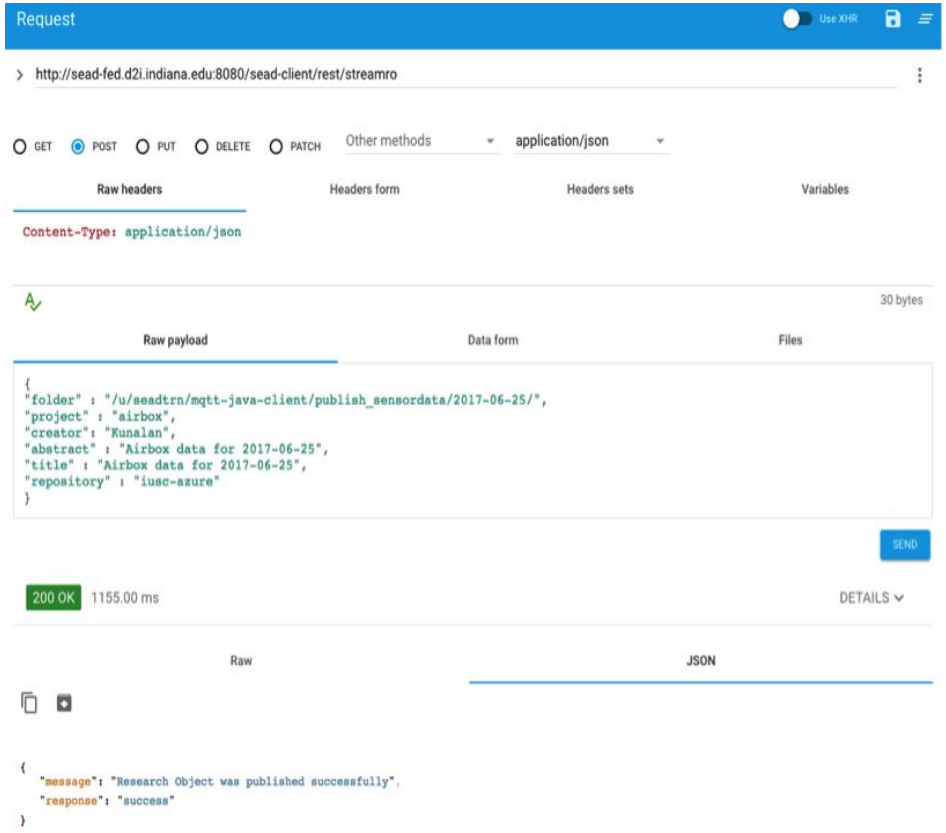
- Copy the data folder for one device/one day into ***publish_sensordata*** folder

```
[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]$
```

1. 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.

2. Later SEAD creates a RO request for daily sensor data, and pass it to SEAD Curbee.

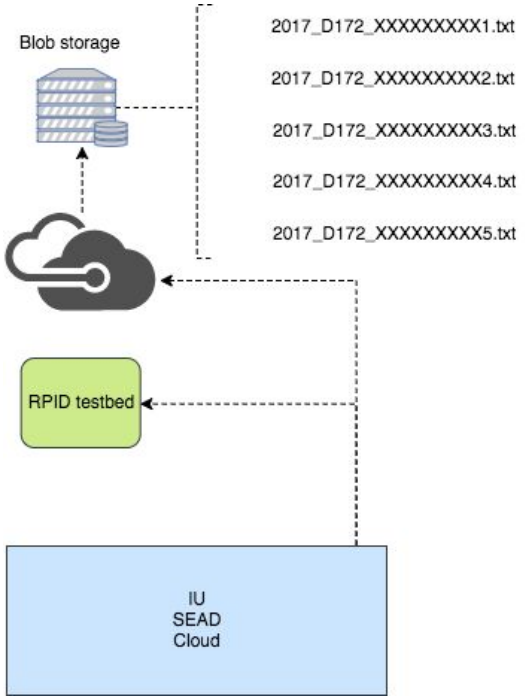
3. SEAD packages raw data files into digital RO.



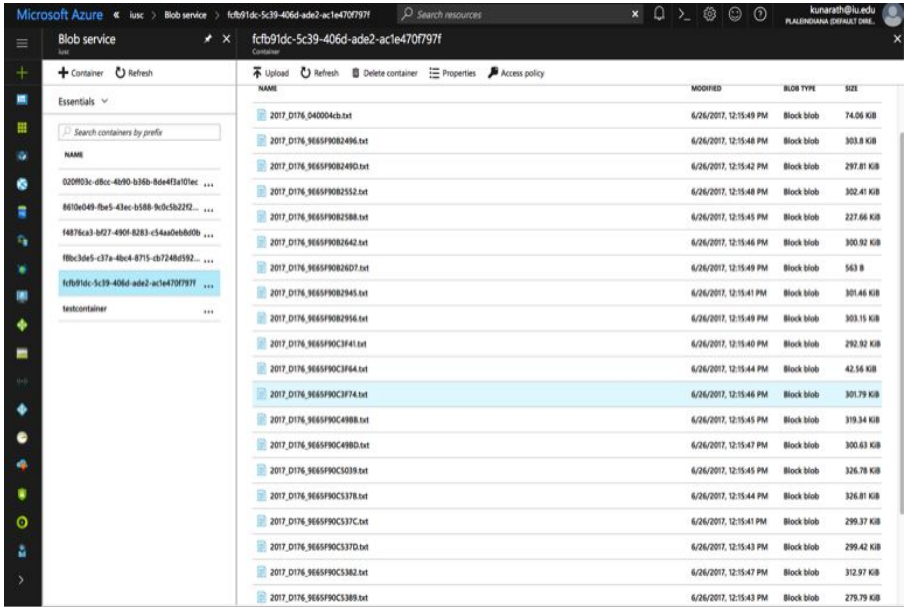
RO ORE map


```
{
  - Aggregation: {
        Creation Date: "2017-06-22T00:00:00Z",
        Last Modified: "2017-06-23T00:00:00Z",
        Identifier: "airbox-67cdd30a-ca9e-48ca-aef7-91afb25158e6",
        @type: "Aggregation",
        Abstract: "Airbox data for 2017-06-22",
        Title: "Airbox data for 2017-06-22",
        @id: "http://d2i-dev.d2i.indiana.edu:8081/sead-c3pr/api/researchobjects/airbox-67cdd30a-ca9e-48ca-aef7-91afb25158e6/oremap#aggregation",
        Creator: "Charitha",
        Publishing Project: "airbox"
    },
    Repository: "iusc-azure",
  + Aggregation Statistics: {…},
    Publication Callback: "http://d2i-dev.d2i.indiana.edu:8080/sead-client/rest/airbox-67cdd30a-ca9e-48ca-aef7-91afb25158e6/status",
  + Preferences: {…},
  + @context: […],
    Rights Holder: "Charitha",
    Affiliations: [ ],
  - Status: [
      - {
            date: "Jun 27, 2017 3:16:43 AM",
            reporter: "SEAD-C3PR",
            stage: "Receipt Acknowledged",
            message: "request recorded and processing will begin"
        },
      - {
            reporter: "iusc-azure",
            stage: "Success",
            message: "http://hdl.handle.net/11723/test.seadtrain.24abbed6-2866-46ea-a6b8-cedaafb40792",
            date: "Jul 5, 2017 12:09:37 PM"
        }
    ]
}
```

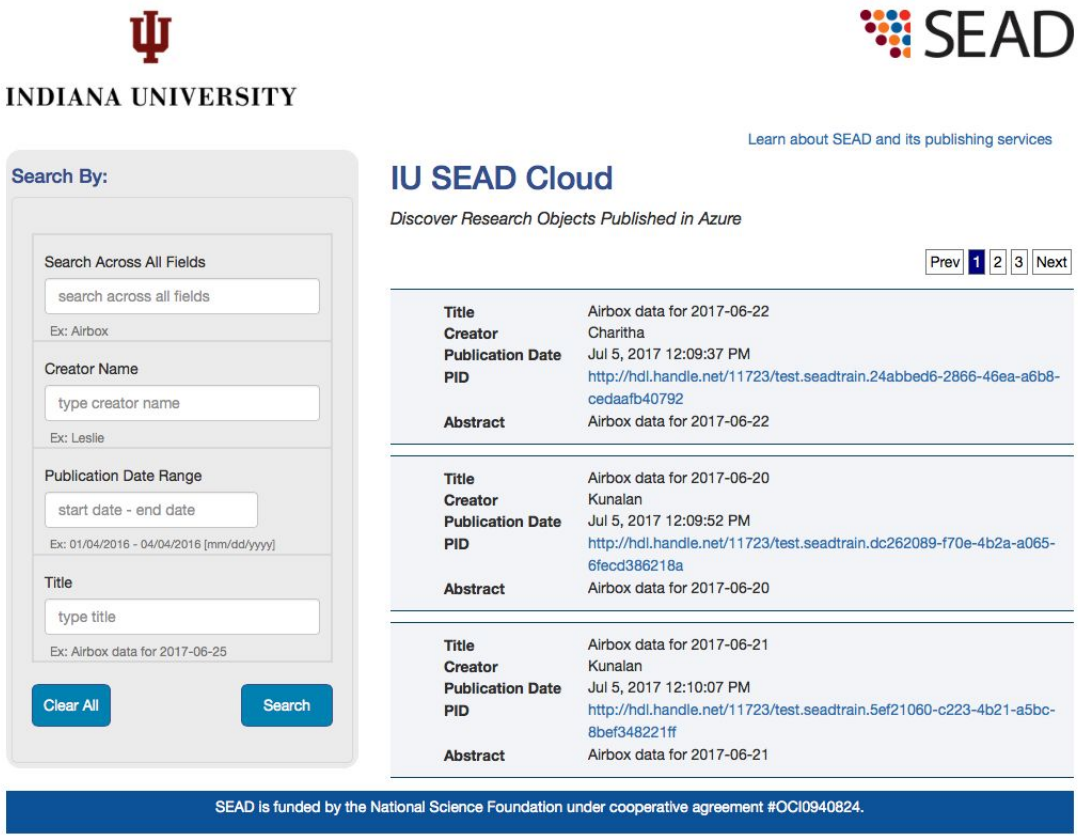
Assign PIDs according to SEAD PID assignment strategy
PIDs minted by handle service of RPID Testbed
Deposit files as BLOBs into Azure repository



After IU SEAD Cloud completes, raw data files are deposited into Azure blob storage



Discovery User Interface for finding published root PID. http://d2i-dev.d2i.indiana.edu:8081/iusc-azure-search/search.html



Next up: Hands-on Setup

- Help Section
  - Hands on participation setup.
  - VMs available on Azure for user to do analysis on raw Airbox sensor data.
  - In folder on VM, there's a folder called "ESIP" that has some scripts in it.
  - Follow the instructions and have fun using Power BI