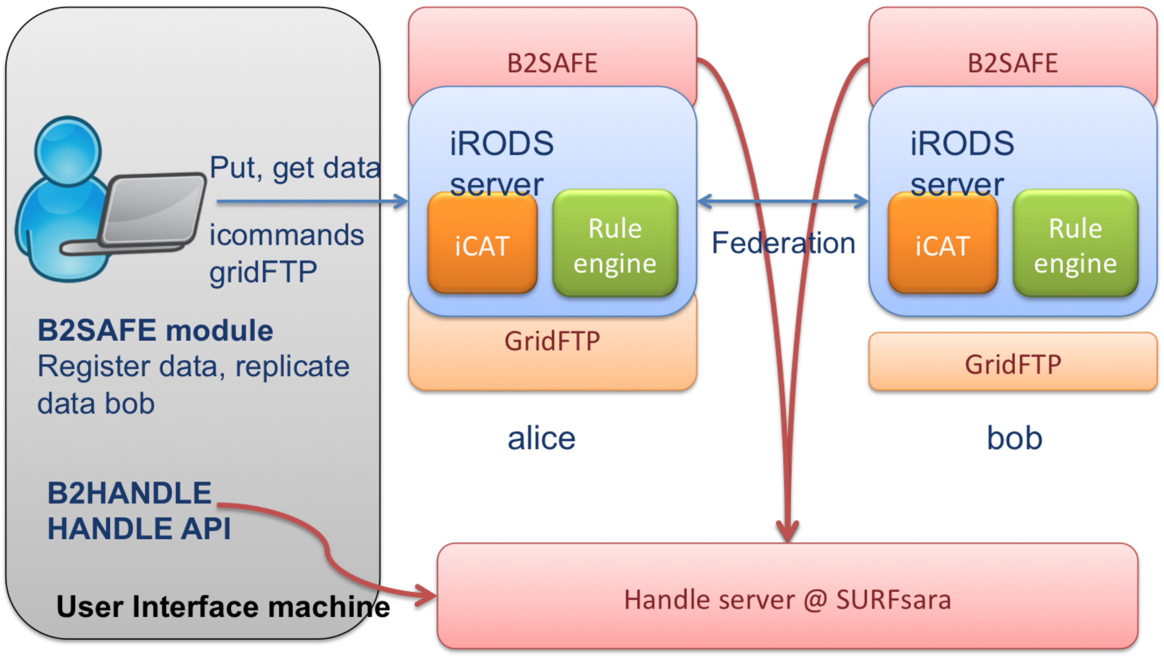
B2SAFE-B2STAGE-B2HANDLE training

Utrecht, January 18th, Christine Staiger (SURFsara)

Training machine - Logins and connections



All information on http://hdl.handle.net/21.T12995/B2SAFE-B2STAGE

User **Interface machine**:
- ssh <user>@130.186.13.170

On the user interface machine you have access to:
- ipython and the B2HANDLE python library
- curl

  → Both needed for the **PID part**
- globus-url-copy and uberftp → needed for the **B2STAGE part**
- icommands → needed for the **B2SAFE part**

All slides, exercises and solutions will be published on:
https://surfdrive.surf.nl/files/index.php/s/6j8SL5Ms9xDqdJK

Handle tutorial

Handle Server responses

1	Success
2	An unexpected error on the server (500 Internal Server Error)
100	Handle not found (404 Not Found)
101	Handle already exists (409 Conflict)
200	Values not found (in resolution, 200 OK; otherwise 400 Bad Request)
201	Value already exists (409 Conflict)
202	Invalid value (400 Bad Request)
301	Server not responsible for handle (400 Bad Request)
402	Authentication needed (401 Unauthorized)

Curl - Cheat sheet

General Handle API definition: http://www.handle.net/tech_manual/HN_Tech_Manual_8.pdf

- Authentication:
  You will find a file '*308_21.T12995_TRAINING_privkey.pem*' and a '*308_21.T12995_TRAINING_certificate_only.pem*' in a folder '*HandleCerts*'.
  Please adjust the variables below and load them in your shell.

```
PRIVKEY=privkey.pem
CERTIFICATE=certificate_only.pem
PID_SERVER=https://epic4.storage.surfsara.nl:8007/api/handles
PREFIX=21.T12995
```

- General header for nearly all requests:

```
-k --key $PRIVKEY --cert $CERTIFICATE \
-H "Content-Type:application/json" \
-H 'Authorization: Handle clientCert="true"' \
```

- Minimal example for **PUT.** Watch out: you need to define 'SUFFIX' first**:**

```
curl -k --key $PRIVKEY --cert $CERTIFICATE \
   -H "Content-Type:application/json" \
   -H 'Authorization: Handle clientCert="true"' \
   -X PUT --data \
      '{"values": [
         {"index":1,"type":"URL",
            "data": {"format": "string",
            "value":"https://ndownloader.figshare.com/files/2292172"}},
         { "index": 100,"type": "HS_ADMIN",
            "data": {"format": "admin",
            "value": {"handle": "0.NA/'$PREFIX'","index": 200,"permissions":
"011111110011"}}}
      ]}' \
$PID_SERVER/$PREFIX/$SUFFIX | python -m json.tool
```

- The construct `$PID_SERVER/$PREFIX/$SUFFIX` can be extended with keywords:

?noredirect	Inhibits resolving to the value stored in ‘URL’, returns always the Handle record
?overwrite=[True, False]	Allow or disable overwriting of the Handle record or parts of its metadata
?index=*n*	Update the n-th index
?<KEYWORD>=<value>	(Only for reverse lookups) Fetches all PIDs where the value of <KEYWORD> equals <value>

- Reverse lookups:
  Assume you know the checksum of a file that was registered under a certain prefix, but you do not know the PID. With the following command you can do a reverse lookup:

```
$PID_REV=https://epic4.storage.surfsara.nl:8007/hrls/handles
curl -k -u "21.T12995:<password>" $PID_REV?<KEYWORD>=*
curl -k -u "21.T12995:<password>" $PID_REV?URL=*\&limit=10
```

CURL - Exercises

1) File and metadata retrieval
    a) How can you retrieve the document behind the PID?
    b) Download the file via the resolver. Try to use *wget* when working remotely on our training machine.
    c) How is the data stored when downloading via the browser and how via *wget*?
    d) How can you retrieve the handle record in a web browser?
    e) Inspect the HS_ADMIN field** at index 100
2) Data and metadata
    a) What happens if you try to reregister the file with the same PID?
    b) Reregister the file with some more metadata:
        - At index 2 insert the key "ORIGIN" with value "Data Carpentry pandas example file"
        - At index 101 insert the key "FORMAT" with value "CSV"
3) Update the Handle record
    a) Create the checksum of the file and store it at index 3 WITHOUT reregistering the file (Hint: use '?index=')
    b) Update the Handle record in one go with
        i) Overwriting index 2
        ii) Creating index 4 with key = "SIZE" and the appropriate value
        iii) Creating index 5 with key = "FORMAT" and value "CSV"
    c) Use curl's DELETE option to delete index 101. Watch out: Do not delete the whole Handle.
4) Linking two files in the Handle layer
    a) Register a local copy of the public file "https://ndownloader.figshare.com/files/2292172"
    b) Store some additional information to check the integrity of the local data
    c) Update the Handle records and mark one of the files as original file and the other as replica
    d) Try to resolve to the local file. Does it work?
5) (Optional) Explore the Handle resolution:
   Create a PID for www.google.com as done above, verify that the resolving works.
   In which of the following cases does the resolving still work and why?
    a) Change the key "URL" at index 1 to *bogus*.
    b) Change the value at index 1 to something that will not resolve.
    c) Insert the key "URL" at some index with the valid URL.
    d) Add the value www.yahoo.com at index 1. Where does the PID resolve to?
    e) Correct the key at index 1 to "URL". Where does the PID resolve to?

B2HANDLE Cheat sheet

General API definition: http://eudat-b2safe.github.io/B2HANDLE/index.html

- Python and ipython commands

ipython	
who	List of all available variables and modules
history	List of all previous commands
Tab completion	Tab completion can be used for - Libraries - Variables - File paths
python	

import <library>
from <library> import <class or function>
	Import libraries and modules

var = <value>	Assigning values to a variable
help(<name>)	The help function. <name> can be - Library - Function - Variable → Gives class definition

- Connect to Handle server
```
cred = PIDClientCredentials.load_from_JSON('<full_path>/<to_crdentials>.json')
ec = EUDATHandleClient.instantiate_with_credentials(cred)
```

- B2HANDLE functions
```
ec.register_handle
ec.modify_handle_value
ec.search_handle
```

B2STAGE tutorial

- Create proxy (execute this command before you start)
  `grid-proxy-init`
- Command to list data on iRODS grid
  `globus-url-copy -list gsiftp://eudat-training2/aliceZone/home/<username>/`
  `uberftp -ls gsiftp://eudat-training2/aliceZone/home/alice/`
- Options for globus-url-copy and uberftp

globus-url-copy	uberftp	
-help	-help	Help
-list	-ls	List directory
	-cat	List contents of file
-p <n>	-parallel <n>	
-r	-r (for deleting, transferring only single files)	Recurse
-cd		Create destination upon transfer
	-mkdir	Create remote directory
-sync, -sync-level		Synchronise data
	-rm (-r), -rmdir	Remove files and folders

- Quick generation of a nested collection on the local filesystem:

```
mkdir -p Data/SubCollection
        for i in {000..002}; do
        echo "Data/File${i} and some text.">"Data/File${i}.txt";
        done
        for i in {003..010}; do
        echo "Data/SubCollection/File${i} and some
text.">"Data/SubCollection/File${i}";
        done
```

B2STAGE exercises

1) Upload of nested collection
   Inspect the help of globus-url-copy. How would you transfer a whole folder?
   a) Create a folder with non-empty data files
   b) Transfer the folder to iRODS
   c) How would you assign a different folder name on the gridFTP server?
   d) Change the content of some files, add some new files in your local data folder
   e) Explore the *-sync* and *-sync-level* options to update your remote data folder
2) Third-party transfer
   You all have access to another gridFTP server which uses the normal linux filesystem.
   ```
   globus-url-copy -list gsiftp://eudat-training3/home/training/
   ```
   a) Create a specific folder for you under this account (you will all use the same account)
   b) Copy your data collection there
   c) Copy the data collection from eudat-training3 to your local home directory
3) Data management by PIDs, the link to B2SAFE
   a) Create a PID for your data collection on aliceZone
   b) The URL needs to have the form
      ```
      irods://130.186.13.14:1247/aliceZone/home/<user>/<collection>/
      ```
   c) Copy the folder to your local home directory using the PID
      ```
      globus-url-copy -list
         gsiftp://eudat-training2/21.T12995/e3d30ab9-ffc2-4b92-87d0-1926d3db9496/
      ```

B2SAFE tutorial

iRODS command cheat sheet

iinit	login
ihelp <command>	Help, if <command> is not specified all commands are listed
ils [-A, -L, -r]	Listing of files and folders - Accession lists - Long format - recursive
imkdir, irm, icp, imv	Like the corresponding unix commands
iput, iget [-r, -K] iput --metadata 'attr1;val1;unit1;attr2;val2;unit2'	Upload and download data to and from iRODS - Recursive - Create/verify checksum
irsync [-r] i:<irods source> i:<irods dest>	Synchronize between two iRODS zones or local file system and iRODS zone
irule [-F, -v]	Execute an iRODS rule - File path - Verbose
imeta - imeta ls [-d, -C] - imeta add [-d, -C] - imeta rm	Create and retrieve metadata from the iCAT catalogue - Get metadata of File (-d) or a Collection (-C) - Create metadata, triple: Attribute, Value, Unit

B2SAFE

You will find example rules in your home folder under *rules/.*
A copy of the B2SAFE rulebase can be found under *rulebase/*.

B2SAFE Exercises

1) **PIDs for Collections**
   Write a rule that takes as argument the path to a collection and creates PIDs for the collection, all subcollections and all files.
   Hint: Explore the file *rulebase/pid-service.re.*
2) **Retrieve PIDs**
   Write a rule that retrieves all PIDs given the collection path.
   Hint: Have a closer look at the function **EUDATiFieldVALUEretrieve**.
3) **Recap: Data transfer between iRODS grids**
   Transfer a data file from */aliceZone/home/<username>* to */bobZone/home/<username>#aliceZone* and introduce a link in the iCAT metadata catalogue so that you can retrieve the copy easily.
4) **Replicating collections with B2SAFE**
   Write an iRODS rule that replicates a data collection from *aliceZone* to *bobZone*.
   a) Inspect the Handle records at http://hdl.handle.net/.
   b) How can you find all replicas of a file or a collection?
   c) When would you use the metadata in the iCAT catalogue and when would you use the information in the Handle system?
5) **Final exercises - Putting it all together**
   a) B2SAFE: Build a chain of three or more replicas. You can also use B2SAFE to replicate to a different iRODS path. What information can you get from the Handle registry and what is stored in the iCAT metadata catalogue?
   b) B2SAFE: Write an iRODS rule to check whether all replicas are still correct.
   c) B2SAFE and the Handle system: Given only the PID to the original file, retrieve all PIDs of the replicas. Use the B2HANDLE python library or the HANDLE REST API.
   d) GridFTP and B2HANDLE: Write a script that up and downloads data to the gridFTP server on *eudat-training3* (this gridFTP server is NOT coupled to an iRODS instance). Upon upload create PIDs. The downloading script should use these PIDs.
   (Example on https://github.com/chStaiger/ELIXIR-gridftp-PID)