Submitter User S3 to S3 Copy Instructions

This Doc is for

1. Setting up a Submitter User controlled AWS S3 bucket for S3 to S3 copying
2. Testing the S3 to S3 copy on our test portal

This Doc will NOT tell you how to

1. Upload to production. However the S3 bucket setup is the same and many of the step are similar.
2. Create a submission user on the test portal

System

macOS Mojave Version 10.14.4
Python 3.6.5 and 3.4.3

Prerequisites:

- AWS Console Account
  - Admin user
  - S3 Bucket with test file
- Python - See requirements file in helper repo
  - awscli
  - encode-utils # Available on pypi and https://github.com/StanfordBioinformatics/encode_utils
  - jsonschema # may be needed.
- Encode Test Portal
  - A submission user on https://test.encodedcc.org
  - Production keys will NOT work
- Optional
  - Helper Repo: https://github.com/ENCODE-DCC/s3tos3-helper

Steps

AWS S3

1. Login to your AWS console account.
2. Create an S3 bucket.
    a. In this example we created a bucket called 's3tos3-copy-test' in 'us-west-2' with recommended public access settings. Our submission bucket is in us-west-2 so we made our test bucket in the same region to limit egress costs.
3. Email encode-help@lists.stanford.edu with the name of your new bucket. Telling us to add your bucket to the s3 to s3 direct copy list. This may take a few days but you can continue with the document. The 'Attempt a copy using your s3 bucket' step near the end will fail however. And you'll need to regenerate user credentials since the expire in 24 hours.
4. Add a test file to the bucket
    a. We uploaded 'encd-test-aws-s32s3.data' file of size 8.0 Bytes for testing.
5. Add the below policy to your bucket after changing the resource name to your bucket. This gives Encoded AWS account access to List and Get objects from your bucket. List is not required but can be helpful for testing.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DelegateS3Access",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::618537831167:root"
            },
            "Action": [
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::s3tos3-copy-test/*",
                "arn:aws:s3:::s3tos3-copy-test"
            ]
        }
    ]
}
```

Setup User Portal Keys

1. Login to the test portal with your submission user email address
    a. https://test.encodedcc.org/
2. Go to you profile
    a. https://test.encodedcc.org/users/7ec6ddd1-1bff-4ac5-b8ac-c677f0ad2d70/
3. Create a new access key.
    a. You can user a previously made key if you have the secret key handy.
4. Export your portal key and secret in your terminal

    a. Helper repo: user-portal-keys.sh - this file must be sourced.

Setup Submission File Object Tsv

1. Create a tsv file with the following two lines. Columns are tab separated.

dataset	file_format	output_type	award	lab	md5sum
ENCSR182HOA	csv	variant calls	U41HG009293	j-michael-cherry	bc50597db0ee5e9d9e79a973f1786345

2. Change the md5sum in red above. Must be unique and 32 alphanumeric characters in length
3. Helper repo: example_submit_file.tsv

Get Submitter AWS credentials via Nathan Watson's encode_utils

1. Source submission vars
    a. Like the portal keys, we'll source a couple vars for convenience
    b. Helper repo: submit-vars.sh - this file must be sourced.
2. Do Dry Run submission

```
eu_register.py -m $TEST_DEMO_URL -p File -i $INPUT_FILE --no-aliases -d
```

    a. See Appendix Dry Run for Error and Success Example Outputs

3. Attempt Submission by removing -d arg from last command

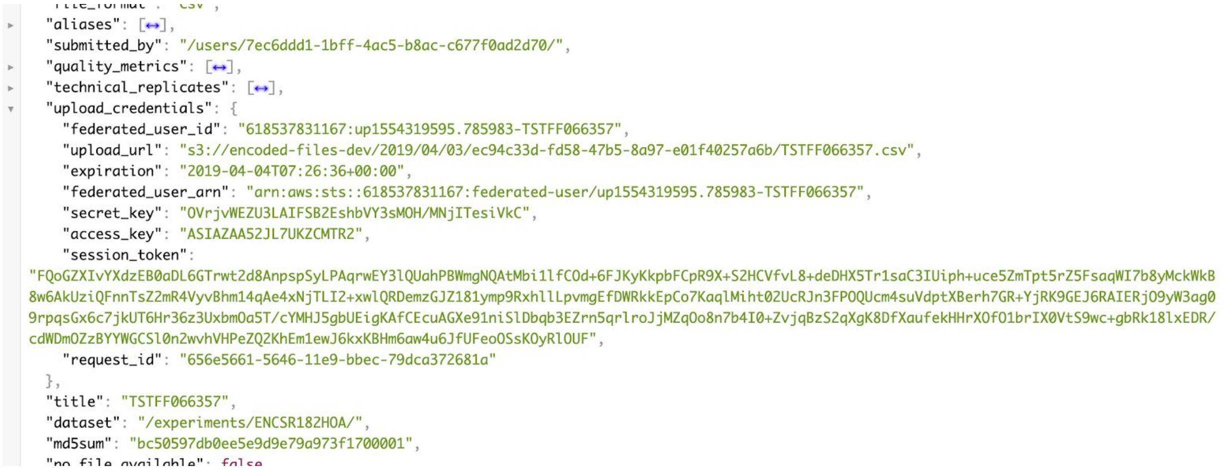
```
eu_register.py -m $TEST_DEMO_URL -p File -i $INPUT_FILE --no-aliases
```

    a. See Appendix Dry Non-Run for Error and Success Example Outputs

4. Get the TST Accession from the Non-Dry Run Success. Blue in the Appendix.
    a. Such as TSTFF066357
5. Go to the test demo using your TST in the following url

https://test.encodedcc.org/files/TSTFF066357/?format=json&datastore=database&frame=object

6. Find the upload_credentials object as seen in image below



```
  "file_format": "csv",
  "aliases": [↔],
  "submitted_by": "/users/7ec6ddd1-1bff-4ac5-b8ac-c677f0ad2d70/",
  "quality_metrics": [↔],
  "technical_replicates": [↔],
  "upload_credentials": {
    "federated_user_id": "618537831167:up1554319595.785983-TSTFF066357",
    "upload_url": "s3://encoded-files-dev/2019/04/03/ec94c33d-fd58-47b5-8a97-e01f40257a6b/TSTFF066357.csv",
    "expiration": "2019-04-04T07:26:36+00:00",
    "federated_user_arn": "arn:aws:sts::618537831167:federated-user/up1554319595.785983-TSTFF066357",
    "secret_key": "OVrjvWEZU3LAIFSB2EshbVY3sMOH/MNjITesiVkC",
    "access_key": "ASIAZAA52JL7UKZCMTR2",
    "session_token":
"FQoGZXIvYXdzEB0aDL6GTrwt2d8AnpspSyLPAqrwEY3lQUahPBWmgNQAtMbi1lfCOd+6FJKyKkpbFCpR9X+S2HCVfvL8+deDHX5Tr1saC3IUiph+uce5ZmTpt5rZ5FsaqWI7b8yMckWkB
8w6AkUziQFnnTsZ2mR4VyvBhm14qAe4xNjTLI2+xwlQRDemzGJZ181ymp9RxhllLpvmgEfDWRkkEpCo7KaqlMiht02UcRJn3FPOQUcm4suVdptXBerh7GR+YjRK9GEJ6RAIERjO9yW3ag0
9rpqsGx6c7jkUT6Hr36z3UxbmOa5T/cYMHJ5gbUEigKAfCEcuAGXe91niSlDbqb3EZrn5qrlroJjMZqOo8n7b4I0+ZvjqBzS2qXgK8DfXaufekHHrXOfO1brIX0VtS9wc+gbRk18lxEDR/
cdWDmOZzBYYWGCSl0n2wvhVHPeZQ2KhEm1ewJ6kxKBHm6aw4u6JfUFeoOSsKOyRlOUF",
    "request_id": "656e5661-5646-11e9-bbec-79dca372681a"
  },
  "title": "TSTFF066357",
  "dataset": "/experiments/ENCSR182HOA/",
  "md5sum": "bc50597db0ee5e9d9e79a973f1700001",
  "no_file_available": false
```

7. We need to add the access_key, secret_key, and session_token for our aws credentials file.
    a. Also note the upload_url as we'll need that later

8. And Create Aws Profile from information.
    a. Put the below block in your Home folder '~/.aws/credentials file.
    b. Create the directory and file if it does not exist.
    c. Do not forget the dot at the beginning of aws.
    d. Please remember these expire in 24 hours

```
[fed-user]
aws_access_key_id=ASIAZAA52JL7UKZCMTR2
aws_secret_access_key=OVrjvWEZU3LAIFSB2EshbVY3sMOH/MNjITesiVkC
aws_session_token=FQoGZXIvYXdzEB0aDL6GTrwt2d8AnpspSyLPAqrwEY3lQUahPBWmgNQAtMbi1IfCOd+6FJKyKkpb
FCpR9X+S2HCVfvL8+deDHX5Tr1saC3IUiph+uce5ZmTpt5rZ5FsaqWI7b8yMckWkB8w6AkUziQFnnTsZ2mR4VyvBhm14
qAe4xNjTLI2+xwlQRDemzGJZ181ymp9RxhllLpvmgEfDWRkkEpCo7KaqIMiht02UcRJn3FPOQUcm4suVdptXBerh7GR+Yj
RK9GEJ6RAIERjO9yW3ag09rpqsGx6c7jkUT6Hr36z3UxbmOa5T/cYMHJ5gbUEigKAfCEcuAGXe91niSlDbqb3EZrn5qrlro
JjMZqOo8n7b4I0+ZvjqBzS2qXgK8DfXaufekHHrXOfO1brIX0VtS9wc+gbRk18lxEDR/cdWDmOZzBYYWGCSl0n2wvhVHP
eZQ2KhEm1ewJ6kxKBHm6aw4u6JfUFeoOSsKOyRlOUF
```

9. Attempt a copy using your s3 bucket, test file, and upload_url we noted above.

```
aws --profile fed-user s3 cp s3://s3tos3-copy-test/encd-test-aws-s32s3.data "paset upload url from above remove quotes"
```

    a. Helper repo: s3tos3-copy.sh

10. A common error is

```
fatal error: An error occurred (403) when calling the HeadObject operation: Forbidden
```

    a. There are a number of reasons this may occur.
        i. Make sure you bucket policy, name, file are all correct.
        ii. You are uploading to the correct bucket and portal. This document is for encoded-file-dev and test https://test.encodedcc.org
        iii. Make sure your credentials have not expired.

If the aws cp fails or any other issues with previous steps, email the help desk with your system info, python version, the output of 'pip list', progress o step that failed, and the output. Also, as silly as this may sound, tell us any steps you may have skipped.

Regenerating Credentials after they have expired

1. Once you've created file upload credentials with eu_register.py you do not need to do it again. A file object already exists in our database so we just need to update it with regenerated credentials.
2. Regenerating credentials can be done with the two links below but may be easier to use the Helper Repo regenerate_creds.py python script.

```
python regenerate_creds.py https://test.encodedcc.org TSTFF066357 -d
```

3. Then remove the -d since that is for a dry run only.
4. Add the new credentials like in the previous section from the url.

https://test.encodedcc.org/files/TSTFF066357/?format=json&datastore=database&frame=object

5. Attempt the aws copy as above too.

Appendix

Dry Run

Error Handling

Bad DEMO_URL

```
ERROR: The specified dcc_mode of 'test.encodedcc.or' is not valid. Should be one of '['prod', 'dev']' or a valid demo.encodedcc.org
hostname.
```

INPUT_FILE cannot be found locally

```
FileNotFoundError: [Errno 2] No such file or directory: 'example_submit_file.ts'
```

Success

```
2019-04-03 10:57:30,995:debug:      Connecting to test.encodedcc.org
2019-04-03 10:57:30,996:debug:      DRY RUN is enabled.
2019-04-03 10:57:30,996:debug:      submission=False: In non-submission mode.
2019-04-03 10:57:30,996:debug:      submission=True: In submission mode.
2019-04-03 10:57:30,997:debug:
IN post().
2019-04-03 10:57:30,997:debug:      <<<<<< POSTING N/A To DCC with URL https://test.encodedcc.org/file and this payload:

{
  "award": "U41HG009293",
  "dataset": "ENCSR182HOA",
  "file_format": "csv",
  "lab": "thomas-gingeras",
  "md5sum": "bc50597db0ee5e9d9e79a973f170001",
  "output_type": "variant calls"
}


2019-04-03 10:57:30,997:debug:      DRY RUN is enabled.
```

Non-Dry Run

Error Handling

Error for incorrect DCC_API_KEY, DCC_SECRET_KEY

```
requests.exceptions.HTTPError: 401 Client Error: Unauthorized for url: https://test.encodedcc.org/file
```

Success - Yes, success ends with 'No file path specified' Exception

```
2019-04-03 12:26:33,114:debug:      Connecting to test.encodedcc.org
2019-04-03 12:26:33,114:debug:      submission=False: In non-submission mode.
2019-04-03 12:26:33,115:debug:      submission=True: In submission mode.
2019-04-03 12:26:33,115:debug:
IN post().
2019-04-03 12:26:33,116:debug:      <<<<<< POSTING N/A To DCC with URL https://test.encodedcc.org/file and this payload:

{
  "award": "U41HG009293",
  "dataset": "ENCSR182HOA",
  "file_format": "csv",
  "lab": "thomas-gingeras",
  "md5sum": "bc50597db0ee5e9d9e79a973f1700001",
  "output_type": "variant calls"
}


2019-04-03 12:26:33,623:debug:          Success.
2019-04-03 12:26:33,624:debug:
IN upload_file()

2019-04-03 12:26:33,624:debug:          >>>>>>GETTING TSTFF066357 From DCC with URL
https://test.encodedcc.org/TSTFF066357/?format=json&datastore=database&frame=object
2019-04-03 12:26:33,820:debug:          >>>>>>GETTING TSTFF066357 From DCC with URL
https://test.encodedcc.org/TSTFF066357/?format=json&datastore=database
Traceback (most recent call last):
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/connection.py", line 1396, in upload_file
    file_path = file_rec[eup.Profile.SUBMITTED_FILE_PROP_NAME]
KeyError: 'submitted_file_name'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/Users/casey/Testing/s3-s3-upload/.venv-s3-to-s3/bin/eu_register.py", line 6, in <module>
    exec(compile(open(__file__).read(), __file__, 'exec'))
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/MetaDataRegistration/eu_register.py", line 282, in <module>
    main()
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/MetaDataRegistration/eu_register.py", line 147, in main
    conn.post(payload, require_aliases=not no_aliases)
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/connection.py", line 927, in post
    self.after_submit_hooks(encid, profile_id, method=self.POST)
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/connection.py", line 649, in after_submit_hooks
    self.after_submit_file_cloud_upload(rec_id, profile_id)
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/connection.py", line 616, in after_submit_file_cloud_upload
    self.upload_file(file_id=rec_id)
  File "/path/to/encode/utiles/repo//utils_encode/encode_utils/connection.py", line 1398, in upload_file
    raise Exception("No file path specified.")
Exception: No file path specified.
```