Integrity Protection for Research Artifacts using Open Science Chain's Command Line Utility

Manu Shantharam
San Diego Supercomputer Center
USA
mshantharam@sdsc.edu

Kai Lin
San Diego Supercomputer Center
USA
klin@sdsc.edu

Scott Sakai
San Diego Supercomputer Center
USA
ssakai@sdsc.edu

Subhashini Sivagnanam
San Diego Supercomputer Center
USA
sivagnan@sdsc.edu

ABSTRACT

Scientific data, its analysis, accuracy, completeness and reproducibility play a vital role in advancing science and engineering. Open Science Chain (OSC) is a cyberinfrastructure platform built using the Hyperledger Fabric (HLF) blockchain technology to address issues related to data reproducibility and accountability in scientific research. OSC preserves integrity of research datasets and enables different research groups to share datasets with the integrity information. Additionally, it enables quick verification of the exact datasets that were used for a particular published research and tracks its provenance.

In this paper, we describe OSC's command line utility that will preserve the integrity of research datasets from within the researchers' environment or from remote systems such as HPC resources or campus clusters used for research. The python-based command line utility can be seamlessly integrated within research workflows and provides an easy way to preserve the integrity of research data in OSC blockchain platform.

ACM Reference Format:
Manu Shantharam, Kai Lin, Scott Sakai, and Subhashini Sivagnanam. 2021. Integrity Protection for Research Artifacts using Open Science Chain's Command Line Utility. In *Practice and Experience in Advanced Research Computing (PEARC '21), July 18–22, 2021, Boston, MA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3437359.3465587

1 INTRODUCTION

The quest for scientific advancement in the fields such as astronomy, medicine, and weather modeling has resulted in a sustained growth of data collection, exploration and analysis. Factors such as the availability of large, complex scientific instruments and sensor networks, and the accessibility of increasingly powerful HPC systems and campus clusters to process, run data-intensive problems, and generate petabytes of data are primary contributors for such sustained growth. Through tests, experiments and hypotheses, researchers generate multiple datasets as byproducts of their own research, and those datasets are then used and cited in other research works. During the research process multiple researchers may work on the same dataset and produce data as part of their research workflow. Investigators working with these evolving data require techniques to check data integrity, track its provenance, as well as provide mechanisms for independent verification for three reasons: (1) to have confidence in the published data / research when building upon prior research, (2) to ensure reproducibility of results, and (3) to ensure uncompromised and accurate data when dealing with a complex workflow with data being generated and moved at various stages of the workflow or in a collaborative environment.

Open Science Chain (OSC). The NSF-funded Open Science Chain (OSC) [4, 9] provides a consortium blockchain platform to store verification information about scientific dataset and provide a unique identifier for the information stored on blockchain. OSC consists of a portal (OSC portal) and a blockchain platform (built using open-source Hyperledger Fabric platform HLF[6]). The OSC portal, integrated with CILogon[7], facilitates metadata contribution and dataset information search through a browser-based interface. Researchers have the ability to contribute information about their dataset including metadata information such as title, description, keywords, DOI, and funding agency, and cryptographic information (SHA256) of their dataset that can be used to verify the authenticity of the dataset by other researchers using the same dataset. OSC does not store actual data but only the metadata and verification information that gets stored as a transaction in the blockchain. The "append" structure of the blockchain prevents altering or deleting previously entered data, allowing the information regarding the dataset to be verifiable and immutable that is essential for reproducibility and audits. OSC's HLF configuration [6] setup includes three distributed peers and orderers, in a "raft" [2] configuration, and a certificate authority to manage private keys and certificates of the identities. Refer [8, 9] for a detailed description of the components, the overall architecture and the workflow of the existing OSC.

At present, the OSC portal is the only way to contribute (add metadata information including cryptographic hash of their data which gets stored as a transaction to blockchain), update (only the user who contributes can update the transaction that gets stored in

PEARC '21, July 18–22, 2021, Boston, MA, USA

ACM ISBN 978-1-4503-8292-2/21/07…$15.00
https://doi.org/10.1145/3437359.3465587

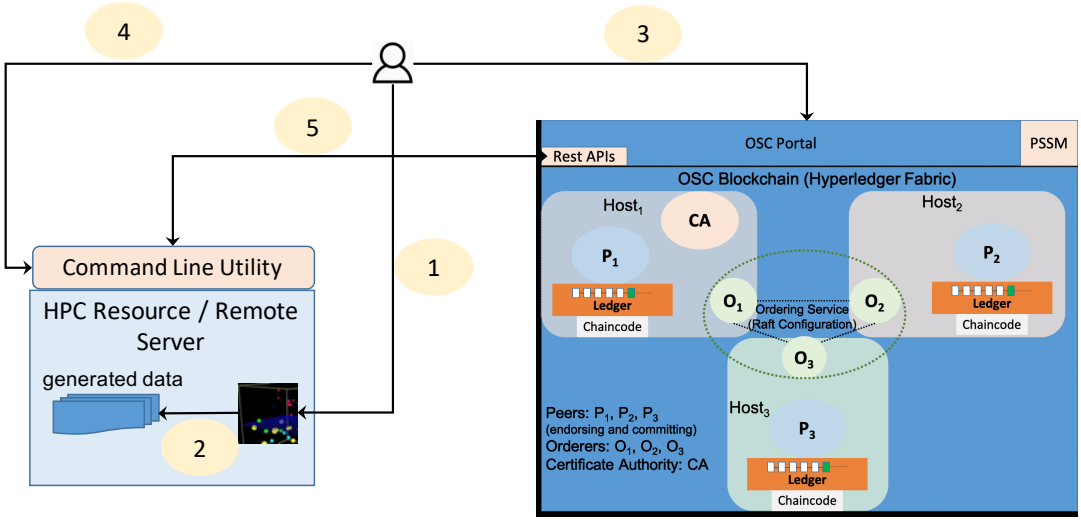

Figure 1: OSC Overview with the Command Line Utility.

the blockchain) or query (search through the entries in blockchain) metadata information. However, to use OSC in its current form:

- The dataset to be contributed should be present on the machine from which the portal is accessed. This requirement is particularly restrictive to enable wide adoption of OSC within HPC and cluster computing community as datasets typically reside on remote systems.
- OSC operations (contribute, update, query) through the portal cannot be automated or embedded within research workflows. Projects with complex data pipelines such as those used in science gateways involve cross-institutional collaborations with a lot of data movement. It is necessary to ensure data integrity and authenticity through an independent, automatic mechanism that blends with the project's data pipeline.
- Large datasets cannot be contributed through the browser, which is typically present in many scientific simulations and data-driven research.

In order to accommodate using OSC from within the researchers' work environment, we have designed and developed a command line utility to perform operations similar to what is supported by OSC portal and provide the ability for automatic data integrity and authenticity verification from remote systems. The remaining sections of the paper describe the Command Line Utility (CLU), its interfaces, current functionalities, and usage with illustrative examples.

2 COMMAND LINE UTILITY (CLU)

We envision the CLU to enable wider adoption of the OSC technology within the data-driven and scientific research communities that work on remote systems such as HPC or campus computing clusters. Typically, users from such communities perform computation, generate and store data within the remote systems. The CLU provides seamless and easy way to contribute this data to OSC without going through the web portal, which has certain limitation as described in the previous section. We have developed a *Python based Command Line Utility* that provides the following functionalities:

- compatible with various operating systems such as Mac/Linux, Windows.
- supports data contribution, updates for data modification as well as query facilities, similar to the ones provided by the web portal.
- provides data contribution with options to include or exclude certain files or directories within a nested directory structure.
- enables large data contributions (greater than 1 GB), which is common within data-driven and scientific research.

Use cases: The following use cases would benefit from having OSC as an independent verification platform to increase the credibility of the research process:

- *Integrity pipeline within HPC systems:* A typical job workflow in an HPC system involves copying and moving data between different storage devices such as local solid state drives (SSDs), scratch and parallel file system during a scientific simulation. Maintaining and monitoring data integrity information manually during various stages of the job pipeline is cumbersome and error prone.
- *Continuous integrity and provenance tracking:* Collaborative data platforms such as SciCrunch [5] or science gateways such as CitSci.org [1] continually update their datasets and analyses. Keeping track of the provenance of datasets helps visualize its evolution and monitoring its integrity information increases trustworthiness of the analyses.

The CLU provides mechanisms to seamlessly integrate OSC with these use cases. Figure 1 provides an overview of the OSC with CLU and illustrates the interactions between the OSC components and its users. The following is an example workflow for contributing or updating metadata to the OSC blockchain using the CLU:

(1) user runs an experiment on a remote system.
(2) the experiment generates data on remote system as part of the researchers' scientific workflow.

```
bash-3.2$ ./osc_client.py --help
usage: osc_client.py [-h] [--template TEMPLATE] [--oscid OSCID] [--token TOKEN] operation

positional arguments:
  operation             'contribute', 'query' or 'update'. For update, first perform
                         a query and then modify the saved yaml file.

optional arguments:
  -h, --help            show this help message and exit
  --template TEMPLATE   template file is mandatory for the contribute and update
                        operations
  --oscid OSCID         osc-id is mandatory for the query operation
  --token TOKEN         pass the authorization key obtained from the OSC Portal.
                        Token is required for contribute and update operations
```

Figure 2: Command Line Utility options.

```
# MANDATORY (One of Files or Directories)
# List of files (start with '- ')
Files:
- /Users/manu1729/sdsc/publications/pearc21/paper.tex

# List of directories (start with '- ').
# Note that all files and directories within the listed directories will be included
Directories:
- /Users/manu1729/sdsc/osc/osc-hfl/config/
- /Users/manu1729/sdsc/osc/OSC-CLI/

# A list of files and directories to exclude during contribution (start with '- ')
ExcludeList:
- /Users/manu1729/sdsc/osc/OSC-CLI/README.txt

# MANDATORY - Title of the contribution
# Include the title after ": " in the same line
Title: testmar182021a

# Description of the contribution
# Include the description after ": " in the same line
Description:

# Keywords for identifying the dataset contribution
# Include a comma separate list of keywords after ": " in the same line
Keywords: data
```

Figure 3: Template for contributing information related to a dataset.

(3) user logs-on to the OSC portal and obtains the authorization token that is used for identification with the CLU. The portal generates a unique per-user, session based token with a preset expiration time.
(4) user provides the token, datasets, metadata information related to the datasets as input to the CLU.
(5) The CLU calls REST APIs to submit relevant information including the per-file SHA256 checksum to the OSC blockchain and provides an appropriate response to the user.

As part of the CLU, we provide a self explanatory template file that can be modified and used for metadata contribution / modification [3]. Figure 2 shows various options available while using the python-based CLU. The *operation* parameter can have one of the values *contribute*, *update*, or *query* corresponding to contribute, update and query OSC operations. The operations *contribute* and *update* can only be performed by an authorized users, whereas a *query* can be performed by anyone interested in browsing the OSC datasets.

2.1 Contributing dataset information

The users of OSC can contribute datasets programmatically using the CLU. Algorithm 1 provides the pseudocode of the contribute operation. The algorithm takes as input the metadata information of the research datasets as well as a list of files to be contributed and the authorization *token*. The data is loaded from the *yaml* file, the integrity information (hash) is computed for the list of files, and all data including the file manifest is converted into the json format acceptable by the OSC REST APIs. The converted data along with the *token* is submitted over an SSL connection to the OSC *Contribute* REST API and its response is stored as a json file within the current working directory. The contributed information can also be viewed on the OSC portal. Users can contribute metadata information of a dataset using the CLU as: *osc_client.py contribute --template file.yaml [--token tok]*, where *--template* takes the name of the file having the metadata information in *yaml* format and *--token* is the command line parameter used to authorize the user (as in step 3 of Figure 1). Figure 3 provides a snapshot of an example contribute *yaml* file. Unlike the OSC portal, the CLU provides options to contribute metadata in terms of both files and directories. The *ExcludeList* is a new feature of the CLU where a user can specify a list of files and directories that have to be excluded from the current set of contributions.

Algorithm 1 Contribute (token, file.yaml)

```
data = yaml.load(file.yaml)
files = getAllFiles(data)
jsonData = convertToJson(data)
manifestList = []
idx = 0
for f ∈ files do
    manifestList[idx] = hash(f)
    idx++
end for
jsonData.append(manifestList)
checkMandatoryFields(jsonData)
contributeData = convertToRESTFormat(jsonData)
responseData = submitData(contributeData, token) {submit the data over SSL using requests.post python API}
saveAsOscId(responseData) {save the response as file with filename as <osc-id>}
```

2.2 Querying and updating dataset Information

The use of query and update operations programmatically enables seamless integration of OSC as an independent data integrity and provenance verification platform within scientific workflows that require frequent data generation and monitoring provenance.

Query. The query operation provides users the ability to search for detailed information such as the metadata and integrity information related to a dataset stored within OSC. We can query a contribution using: *osc_client.py query --oscid osc-id*, where *--oscid* takes the *oscid*, the OSC specific unique identifier of the contribution. Further, a query is used during the process of updating a contribution. Once a user executes a query operation with a valid *osc-id*, the corresponding contribution is stored as an easy-to-ready yaml file which can be used to update the contribution. The query will also be expanded to include email address as a search option.

Update. The datasets used in research evolve over time as new experiments / algorithms use existing data and produce new data. Users can update the metadata information of this data or modify the list of files using the *update* operation. This can be achieved using the CLU: 1) query the original contribution using the *osc-id* that stores the response as an yaml file, and 2) modify the yaml file appropriately and then use the following to perform an update operation: *osc_client.py update --template file.yaml [--token tok]*, where *--template* takes the name of the generated *yaml* file. The functionality of the *update* operation is very similar to the *contribute* operation, except: it (1) updates an existing contribution and (2) stores the *diff* of the original and the updated list of files in the current working directory, i.e., records the list of new, updated, deleted, and unmodified files compared to the previous contribution in a *changes.txt* file. An *update* operation is appended to the OSC blockchain while keeping a trail of all previous transactions capturing provenance of changes related to the original contribution.

The described CLU process can be integrated into researchers' workflow on remote systems and enable storing the integrity information of the scientific artifacts used in the research in a blockchain from within the researchers' environment. Based on researchers' feedback, we will further refine CLU and support other metadata entities that might be of interest to the scientific community.

3 CONCLUSION

We describe the OSC Command Line Utility (CLU), a Python based command line tool to enable researchers to contribute, modify or query information about artifacts within OSC from the researchers' work environment including remote systems such as HPC and campus clusters. We envision the CLU to lower the complexity barrier for the use of OSC as a platform for independent verification of authenticity and integrity of scientific data, and to promote adoption within the research community that use remote systems for compute, storage and data analysis. Further, the CLU provides the ability to perform automatic data integrity and authenticity checks that could be seamlessly integrated with projects involving workflows with complex data pipelines. As part of our ongoing and future work, we will be working with a few science gateways such as CitSci.org and end users of HPC resources (e.g. Expanse at SDSC) to help incorporate CLU into their research workflows.

ACKNOWLEDGMENTS

Open Science Chain is supported by the National Science Foundation under Award Number 1840218.

REFERENCES

[1] [n.d.]. *CitSci.org*. https://citsci.org
[2] [n.d.]. *HLF raft Protocol*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/orderer/ordering_service.html#raft?
[3] [n.d.]. *Open Science Chain Git Repository*. https://github.com/OpenScienceChain
[4] [n.d.]. *OSC*. https://www.opensciencechain.org
[5] [n.d.]. *SciCrunch Infrastructure*. https://scicrunch.org
[6] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolic, Sharon Weed Cocco, and Jason Yellick. 2018. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *EuroSys 2018*. Association for Computing Machinery,, New York, NY, 1?15.
[7] J. Basney, H. Flanagan, T. Fleury, J. Gaynor, S. Koranda, and B. Oshrin. 2019. CILogon: Enabling Federated Identity and Access Management for Scientific Collaborations. In *Proceedings of the International Symposium on Grids and Clouds (ISGC), PoS(ISGC2019)031*. https://doi.org/10.22323/1.351.0031
[8] Manu Shantharam, Scott Sakai, Kai Lin, and Subhashini Sivagnanam. 2020. Towards building a Fault Tolerant and Secure Open Science Chain. In *Gateways 2020 Posters*. Virtual.
[9] S. Sivagnanam, V. Nandigam, and K. Lin. 2019. Introducing the Open Science Chain – Protecting Integrity and Provenance of Research Data. In *PEARC19 Proceedings*. Chicago, IL.