PROCEEDINGS OF SCIENCE

Blockchain-Enabled Secure Management of Scientific Data in Permissioned Networks: a Metadata-Centric Approach

Domingo Ranieri,[a,*] **Alessandro Costantini**[a] **and Barbara Martelli**[a]

[a] *INFN-CNAF, Viale Carlo Berti Pichat 6, 40127 Bologna, Italy*
E-mail: domingo.ranieri@cnaf.infn.it, alessandro.costantini@cnaf.infn.it, barbara.martelli@cnaf.infn.it

In recent years, blockchain has emerged as a promising new technology to manage trusted information, making it easier for companies to access and use critical data while maintaining the security of this information.
Permissioned blockchains, unlike permissionless ones, restrict access to a select group of certified entities. They ensure a controlled and secure environment where only authorized participants can join the network and perform operations, a peculiar aspect in sectors where data sensitivity, confidentiality, and limited access are crucial.
Tracking operations performed on the data and guaranteeing reproducibility of research through workflow reconstruction upon data processing become very important in different sectors ranging from scientific communities to private companies and health.
This is the case of the present activity, where the implementation of a permissioned blockchain system aimed at ensuring data immutability, operations traceability, and the ability to reproduce workflows is presented and discussed. In such regards, we work with Hyperledger Fabric, an enterprise-grade permissioned distributed ledger platform that offers modularity and versatility for a broad set of industry use cases.

International Symposium on Grids and Clouds (ISGC2024)
24 -29 March, 2024
Academia Sinica Computing Centre (ASGC), Institute of Physics, Academia Sinica
Taipei, Taiwan

[*] Speaker

PoS(ISGC2024)043

1. Introduction

In scientific research, the integrity and reproducibility of data are paramount for ensuring the reliability and credibility of research findings. However, the increasingly complex nature of scientific data, data management and transparency challenges pose significant efforts to achieving these goals.

One of the primary challenges in scientific data management is ensuring data integrity and workflow reproducibility throughout its lifecycle. Researchers frequently face challenges in authenticating data, tracing its origins, and replicating experimental workflows, raising concerns regarding data reliability and the outcomes of research endeavours. In response to these challenges, our proposed solution aims to address the following objectives:

- Ensure the integrity of scientific data by implementing robust mechanisms for data validation and tamper-proof record-keeping.
- Enhance the traceability of data operations, enabling researchers to track the history of data modifications and ensure transparency and accountability.
- Facilitate the reproducibility of experimental workflows by recording metadata attributes such as processing algorithms and versioning information.
- Provide researchers and organizations with a secure and transparent platform for managing scientific data, fostering collaboration and innovation in scientific research endeavours.

The National Institute of Nuclear Physics (INFN [1]), in collaboration with the National Research Centre for High-Performance Computing, Big Data and Quantum Computing (ICSC [2]), started an investigation to implement a system aimed at ensuring data immutability, operations traceability and workflow reproducibility by leveraging blockchain technology and innovative data management techniques.

The work presented in this paper has been funded by the NextGenerationEU European initiative through the Italian Ministry of University and Research, PNRR Mission 4, Component 2 - Investment 1.4, Project ICSC, code CN00000013 - CUP I53C21000340006.

This paper has the following structure: Section 2 introduces blockchain technology. Section 3 describes the technologies and solutions adopted. Section 4 outlines the architecture of the developed service. Section 5 discusses data management, and Section 6 delves into the code and its underlying mechanics. Finally, Section 7 offers conclusions and considers future developments.

2. Blockchain technology

Blockchain technology emerged with the creation of Bitcoin [3] in 2008 by an anonymous person or group known as Satoshi Nakamoto. Bitcoin introduced the concept of a decentralized digital currency, enabled by a revolutionary technology called blockchain.

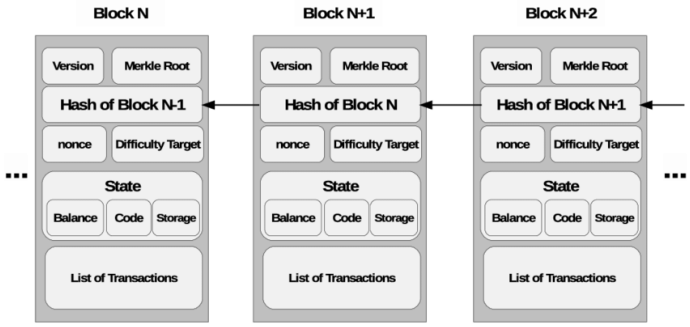

Figure 1: Blockchain schema

2.1 Characteristics of Blockchain

This technology has since evolved beyond its initial application in cryptocurrencies. It is a distributed ledger technology which enables secure, transparent, and decentralized record-keeping of transactions across a network of computers (nodes). Blockchain operates on a decentralized network of computers, where data is distributed and synchronized across multiple nodes. This decentralization enhances the resilience and security of the system, as there is no single point of control or failure. Decentralization also promotes trust and collaboration among researchers, as it eliminates the need for intermediaries and reduces the risk of data manipulation or censorship.

Data recorded on the blockchain are immutable and tamper-proof. Each block in the blockchain is cryptographically linked to the previous block, creating a chain of blocks (Figure 1).

Only the consensus of the majority of the network participants can modify this chain of blocks. By leveraging blockchain technology, we can manage data integrity and immutability.

Blockchain networks rely on consensus mechanisms to reach agreement among nodes on the validity of transactions and the estate of the ledger. Common consensus mechanisms include Proof of Work (PoW) [3], Proof of Stake (PoS) [4], and Practical Byzantine Fault Tolerance [5] (PBFT). These mechanisms ensure that all nodes in the network reach consensus on the order and validity of transactions, thereby maintaining the integrity of the blockchain.

Blockchain technology provides a transparent and auditable record of all transactions and changes made to the data. Each transaction is timestamped and linked to previous transactions, creating a transparent and immutable audit trail. It enables researchers and stakeholders to track the provenance of data, verify its authenticity, and ensure compliance with regulatory requirements.

2.2 Permissionless vs. Permissioned Blockchains

Blockchain can be divided into permissionless or public and permissioned or private. Permissionless blockchains operate on an open participation model, allowing anyone to join the network without requiring permission or approval. Participation is decentralized, with consensus mechanisms such as PoW or PoS used to achieve agreement among participants. Permissionless blockchains prioritize transparency and immutability, with all transaction data being publicly accessible and recorded on the blockchain in a tamper-proof manner. Examples of permissionless blockchains include Bitcoin and Ethereum [6].

On the other hand, permissioned blockchains are characterized by restricted access, typically requiring permission or approval from a central authority or consortium to participate in the network. Permissioned blockchains prioritize privacy and confidentiality, offering features such as private channels and selective data sharing to ensure sensitive information remains protected. Examples of permissioned blockchains include Hyperledger Fabric [7].

3. Technologies adopted

In the present activity, a Proof of Concept (PoC) utilizing Hyperledger Fabric to establish a private blockchain network has been developed and deployed. The choice of a permissioned blockchain for our work on scientific data management reflects the need for enhanced privacy, access control, regulatory compliance, and scalability, which are fundamental to managing scientific data effectively and securely.

In the present section, a high-level description of the technologies adopted is depicted.

3.1 Hyperledger Fabric

Hyperledger Fabric has been selected as a blockchain framework implementation for its enterprise-grade capabilities, permissioned network model, and privacy features, which align well with the requirements of your PoC solution for scientific data management.

Hyperledger Fabric provides a permissioned blockchain framework, allowing you to define membership criteria and access controls. This feature ensures that only authorized participants can join the network, enhancing security and privacy.

Fabric's modular architecture separates components such as consensus, membership services, and smart contract execution. This modularity enables flexibility and customization, allowing you to tailor the blockchain network to specific use cases and requirements.

Fabric provides a REST API, allowing easy integration with external systems and applications. This API enables seamless communication between the blockchain network and client applications, facilitating interoperability and integration with existing systems.

Fabric supports smart contract development in multiple programming languages, including Go [8], JavaScript [9], and Java [10].

3.2 Flask

Flask [11] is a Python web framework which provides tools and libraries to help developers build web applications and APIs in Python. Flask functions as the client-side interface for interacting with the blockchain network. It is highly flexible and extensible, allowing you to integrate additional libraries and extensions as needed. These features make it ideal for rapid prototyping and development of Proof of Concept projects.

Flask has been used to develop a client responsible for managing data and metadata storage. Flask serves as the interface between the users and the blockchain network, providing data submission functionalities, retrieval, and management. Additionally, the Flask client saves data locally and sends metadata to the blockchain network through APIs. This design enables flexibility in data storage, allowing users to choose the most suitable storage system for their needs. The Flask

client can be easily modified to integrate with different storage systems, ensuring compatibility and adaptability to evolving requirements.

3.3 Indigo-IAM

Authentication and authorization within our solution are provided by Indigo-IAM [12], an Identity and Access Management system, which follows OpenID Connect (OIDC) [13]/OAuth 2.0 [14] protocols developed within INFN.

Indigo-IAM system ensures that only authenticated and authorized users can access the blockchain network and interact with the stored data and metadata. The Flask client authenticates users through the OIDC protocol provided by the private IAM, ensuring secure access to the system. The Flask client also utilizes OAuth 2.0 for authorization, allowing users to perform authorized actions based on their roles and permissions.

3.4 MongoDB

In our PoC, MongoDB [15] has been used as an off-chain storage system to accommodate the storage needs of the service. MongoDB is a NoSQL database known for its flexibility, scalability, and high performance, making it suitable for handling structured and unstructured data. It utilizes a document-oriented approach to store data in JSON-like formats, which offers flexibility over traditional relational database schemas. This model simplifies data integration for various applications and enhances developer productivity by aligning with modern, document-based structures used in web and mobile applications. In our PoC, MongoDB plays a critical role in handling off-chain data storage. It manages the storage of voluminous scientific datasets impractical to store on the blockchain, ensuring data accessibility and integrity without compromising performance.

Modifying the client's backend, it can interact with any storage system, including SQL databases, data lakes or cloud storage platforms, leaving the interaction with the blockchain unchanged. By decoupling metadata management from underlying storage systems, our service ensures compatibility and interoperability between different storage solutions while maintaining the integrity and security of data on the blockchain.

4. Architecture of the service

The architecture of the service is designed to manage scientific data using blockchain technology. It consists of three main components: the blockchain network, the Flask client, and the off-chain storage system. The blockchain network, powered by Hyperledger Fabric, provides a decentralized ledger for recording transactions and managing metadata. The Flask client functions as the user interface, enabling users to interact with the blockchain network, authenticate, and perform data management tasks. The off-chain storage system, provided by MongoDB, complements the system by offering flexible storage solutions.

4.1 Blockchain Network

Within the blockchain network (see Figure 2),are instantiated two peers and an orderer node to enhance fault tolerance and ensure redundancy in case of node failures. This configuration provides network stability and reliability, even during unexpected disruptions.

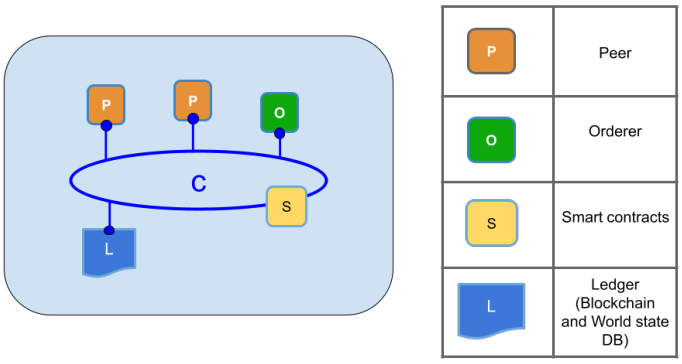

Figure 2: Blockchain network architecture implemented in the PoC.

The Certification Authority provided by Hyperledger Fabric has been used to generate cryptographic materials, including certificates and private keys, for each organization and node within the network.

The network follows a Raft [16] consensus algorithm, one of the consensus mechanisms supported by Hyperledger Fabric. Raft consensus ensures a quorum among the nodes to agree on transactions, maintaining the integrity and consistency of the blockchain ledger.

Furthermore, we defined a channel for communication and consensus between the nodes. The channel contains a ledger that serves as the backbone of the blockchain network. This ledger is composed of the blockchain itself and a world-state database. The latter represents the current state of the data and assets stored on the network, allowing for efficient querying and access to the latest data without requiring traversal of the entire blockchain. Smart contracts, written in Go, are deployed to define the structure and logic for managing metadata objects and operation information. The smart contract defines two structs: *Metadata* (1) and *OperationInformation* (2), along with functions to operate with them.

To interact with the blockchain network, Hyperledger Fabric made available a REST API interface that allows external clients to communicate with the network and execute functions defined in the smart contract.

4.2 Flask Client

The Flask client serves as the user interface for interacting with the blockchain network and managing data and metadata. It provides a simple web application that allows users to log in using Indigo-IAM. Upon successful authentication, users receive a JSON Web Token (JWT) [17] containing information about their identity and authorization groups that authorize them to perform specific operations within the system.

The Flask client provides a user-friendly interface with multiple pages to save and retrieve information from the blockchain. These pages are accessible to logged-in users and provide insights into the data stored on the blockchain. Users can navigate among different pages to view metadata associated with their account or specific datasets.

The client provides forms and input fields for users to input data. Upon receiving new data, the Flask client contacts the off-chain storage system through APIs to save the data, extract and process

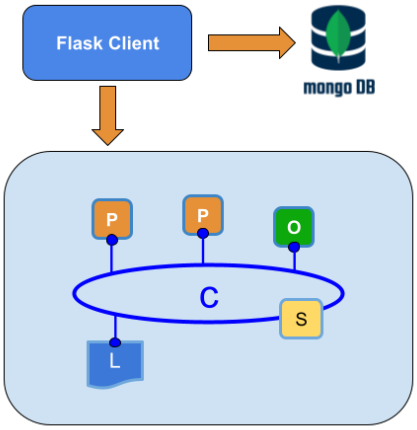

Figure 3: Service Architecture

metadata, and store them in the blockchain with information about the operation performed. This integration enables seamless interaction between the Flask client, the blockchain network, and the off-chain storage system, facilitating efficient data management and workflow automation.

4.3 Off-chian Storage system

The integration between MongoDB and our system is facilitated through a well-defined API, which the Flask client utilizes to interact with the database. The client acts as the intermediary, capturing user inputs and system-generated data and securely transmitting this information to MongoDB.

Once the data reaches MongoDB, it is stored in a structured format that supports high efficiency and easy retrieval. This setup allows our system to quickly access and manipulate stored data, catering to real-time processing needs and complex queries. The use of MongoDB enhances the scalability of our data storage solution and ensures that the data is readily accessible without overwhelming the blockchain network.

5. Data management

In this section, we explore the fundamental aspects of data management within our system, emphasizing the organization and handling of data and metadata to enhance operational efficiency and data integrity. By focusing on the distinction between data and metadata, the extraction and storage of metadata, and the execution of data operations, our system manages data effectively and aligns with the best practices for data security and regulatory compliance. The following subsections detail specific techniques and methodologies employed to achieve these aims.

5.1 Data vs. Metadata

Data refers to the actual content or information being stored, such as measurements, observations, or experimental results.

Metadata provides additional information about the data, serving as a descriptor or tag that enhances understanding and context. While technical metadata may contain experiment-specific details like instrument used or measurement units, we focus primarily on metadata that ensures data validity and workflow reconstruction.

Metadata improves searchability by enabling users to efficiently locate specific datasets based on metadata attributes, such as owner, timestamp, and type.

By including key metadata elements like hash, data usability is enhanced by providing mechanisms for data validation, immutability verification, and workflow reconstruction.

In our scientific data management system, we decided to save metadata on the blockchain while keeping the actual data off-chain. This approach is motivated by two primary reasons:

1. **Efficient Utilization of Blockchain Storage Space:** Blockchain storage space is a finite and ever-increasing resource. Therefore, it is crucial to manage it judiciously to minimize the rate of increase and ensure long-term sustainability. By saving only relevant metadata on the blockchain, we optimize storage utilization and mitigate the risk of excessive blockchain bloating. Storing entire datasets on the blockchain would lead to rapid consumption of storage resources, potentially limiting the scalability and performance of the system.

2. **Preservation of Data Privacy and Security:** Data privacy and security are paramount considerations in scientific data management. Storing sensitive or private data directly on the blockchain poses significant risks, as blockchain data is inherently accessible to all participants in the network and cannot be deleted or modified easily. This lack of data privacy and immutability can be problematic, especially when dealing with confidential or proprietary information. By keeping the actual data off-chain, we maintain greater control over access permissions and ensure compliance with data protection regulations.

Saving only metadata on the blockchain, it gains efficiency, scalability, and data privacy, enabling secure and transparent management of scientific data while leveraging the benefits of blockchain technology.

5.2 Metadata Extraction and Storage

When operating with data, we have to deal with two different definitions: raw data and processed data. While raw data refers to the original, unprocessed data collected from experiments, observations, or measurements, processed data results from the analysis or manipulation of raw data to extract meaningful insights, derive new information, or perform calculations. Documenting information about the production of processed datasets is critical for ensuring reproducibility and transparency in data analysis. Including details such as the algorithms used, parameter settings, and software versions provides valuable insights about processed dataset generation. Additionally, saving metadata such as the hash of the code or links to the code repository enables other researchers to replicate the data processing steps and validate the results independently.

From such principles, metadata extraction and storage play a pivotal role in our system. A metadata struct that encapsulates essential attributes capturing relevant information about datasets has been defined and implemented as a smart contract. Hereafter, we report the metadata struct we defined, presenting and describing its components.

Listing 1: Metadata structure

```
type Metadata struct {
        ID              string `json:"ID"`
        DatasetName     string `json:"DatasetName"`
        Owner           string `json:"Owner"`
        FromHash        string `json:"FromHash"`
        Version         string `json:"Version"`
        Algorithm       string `json:"Algorithm"`
        Timestamp       string `json:"Timestamp"`
}
```

- **ID:** The hash of the dataset, serving as a unique identifier.
- **DatasetName:** A human-readable name for the dataset.
- **Owner:** The entity or individual who owns the dataset.
- **FromHash:** The hash of the dataset that the current dataset is pointing to, facilitating workflow reconstruction and versioning.
- **Version:** A numerical value automatically incremented to denote the version of the dataset.
- **Algorithm:** A reference to the algorithm used for dataset processing, which can be a link to the algorithm code repository. If the algorithm is proprietary and private, this voice can be its hash.
- **Timestamp:** The timestamp indicating the creation time of the dataset.

In our web application, users can navigate to the **Record Raw dataset** or **Record Processed dataset** page to input various details.

In the **Record Raw dataset** page, users provide the DatasetName and upload the raw data. If the dataset is associated with another dataset, users also input the hash of the related dataset. The client uses these inputs to generate the information sent to the blockchain. Upon submission, the client calculates the hash of the raw data, which serves as the unique identifier (ID) for the metadata saved in the blockchain. The client extracts the dataset owner from the JWT token provided by the user and if a related dataset hash is provided, it stores in the **FromHash** attribute. The client also calculates the **Version** and the **Timestamp**.

In the **Record Processed dataset** page, users can provide details about the new dataset being added, such as the hash of the processed dataset, the DatasetName, and information about the algorithm used in processing the dataset. This information could be the code hashed or a link to the repository where the code is stored. Similar to the previous case, these inputs are processed by the client and sent to the blockchain. The client calculates the hash of the new dataset to serve as metadata **ID** and passes the related dataset's hash in the **FromHash** attribute. Notably, in this case, the **Algorithm** attribute is initialized with the details provided by the user regarding the processing algorithm used for the dataset.

Additionally, the client calculates the dataset hash if not provided manually. Once the client extracts metadata and validates it to meet specific criteria, such as format and completeness, it passes to the blockchain for storage using APIs. This operation triggers the execution of code in our

smart contract that defines the logic for processing incoming transactions and recording metadata on the blockchain ledger. Then, the client can save the dataset in the off-chain storage system if provided.

5.3 Data Operations

Each operation (such as recording a raw or processed dataset) involves essential information extraction and recording. This information, known as Operation Information, is passed to the blockchain via APIs. In the smart contract, an OperationInformation struct has been defined and used to create an asset stored in the blockchain using information provided by the client.

Listing 2: Operation Information structure

```
type OperationInformation struct {
        ID             string `json:"ID"`
        Operation_type string `json:"Operation_type"`
        Executed_by    string `json:"Executed_by"`
        Timestamp      string `json:"Timestamp"`
}
```

- **ID:** The ID of the operation, is the hash of the Metadata ID key.
- **Operation_type:** Specifies the type of operation performed, such as recording a raw dataset or processing a dataset.
- **Executed_by:** Indicates the entity or individual who executed the operation, initialized using the ID extracted from the JSON Web Token (JWT) obtained during authentication.
- **Timestamp:** Records the time at which the operation was executed.

Operation Information documents and tracks the execution of various operations within the system. Recording details, such as the operation type, executor, and timestamp, provides valuable insights into the history and provenance of data management activities.

6. Objective Fulfillment

In this section, we examine how our system achieves its designated goals through the implemented functionalities designed to boost operational effectiveness. Concentrating on critical elements such as data validation, workflow reconstruction, and transparency, we ensure that our system adheres to industry standards for security and interoperability. The following subsections will elaborate on the mechanisms employed to fulfil these objectives.

6.1 Data validation

The validity of datasets is ensured by saving the hash of the dataset in the blockchain. This hash serves as a cryptographic fingerprint of the dataset, guaranteeing its integrity and immutability. To verify the validity of an off-chain dataset, users can calculate its hash and compare it with the hash saved in the blockchain. Additionally, our system provides a search page that allows users to retrieve metadata associated with a specific dataset hash. By inputting the hash into the search page, users

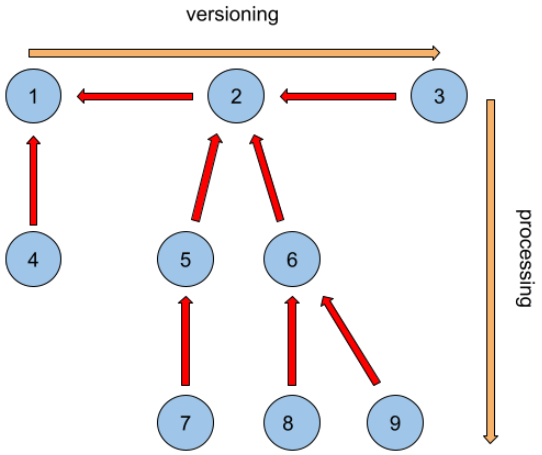

Figure 4: Example of DAG

can query the blockchain network and retrieve the corresponding metadata if it exists. This process facilitates easy validation and verification of datasets, enabling users to confirm the authenticity and integrity of their data through blockchain-based verification mechanisms. It enhances trust and reliability in the system, ensuring that scientific data remains secure and tamper-proof throughout its lifecycle.

When users search for metadata using their hash from the client interface, the system retrieves the metadata related to the dataset and the information related to its creation. This integration enhances the transparency and traceability of data management processes, enabling users to reconstruct the context proper of the datasets.

6.2 Workflow recostruction

The workflow reconstruction within our system is facilitated by specific attributes:**FromHash** and **Algorithm**. Together, these attributes enable the construction of a Directed Acyclic Graph (DAG) representing the data workflow. Each node in the graph corresponds to a dataset, identified by its hash value. These nodes are interconnected by arrows that point from one node to another with a hash value matching the **FromHash** attribute of the starting node. Additionally, the **Algorithm** attribute provides insights into the processing steps employed to generate each dataset. By leveraging this information, users can reconstruct the data workflow, trace the lineage of datasets, and understand the processing methods applied at each stage. We report an example in Figure 4.

In this case, node n.1 is the first raw dataset uploaded. We organized the DAG to have the raw dataset versioning in the first row from left to right. In this way, each dataset in the first row points from right to left. The processed datasets are the nodes n.4-9. We also highlight the possibility of performing processing on processed datasets.

6.3 Transparency and interoperability

About interoperability, since our system does not store sensitive data on the blockchain, the information saved is accessible to anyone who can access the blockchain network. This transparency

enables researchers to read metadata stored on the blockchain. Furthermore, researchers can produce processed or new versions of existing datasets and store metadata on the blockchain, pointing to the hash of the previously recorded dataset. This capability facilitates collaboration and promotes the reproducibility of research findings within the scientific community. It is worth mentioning here that access to specific operations, including reading information saved on the blockchain, can be restricted based on user permissions coded in the JSON Web Token (JWT). Such functionality may allow organizations to define and enforce fine-grained permissions tailored to their specific requirements, enhancing data security and ensuring regulatory compliance in scientific data management practices.

7. Conclusions

In the present contribution, a PoC leveraging blockchain technology to ensure data immutability, operation traceability, and workflow reproducibility in scientific data management has been presented and described.

Data validity is ensured in our system using cryptographic hashing techniques applied to the dataset's metadata, which includes a hash value stored on the blockchain services as a unique identifier for the dataset and used to verify its integrity. By comparing the calculated hash of a dataset with the hash stored on the blockchain, users can confirm the data and detect any tampering or modification on it.

Operation traceability has been facilitated by recording operation information on the blockchain after every data management operation, including details such as the type of operation performed, the entity executing the operation, and the operation timestamp. By storing this information on the blockchain, users can trace the history of operations performed on the data, ensuring transparency and accountability throughout the data lifecycle.

Last but not least, workflow reproducibility is provided by recording metadata attributes to enable the reconstruction of data lineage and workflow from one side and getting information about the algorithms or processes used to generate or process the data from the other.

By saving this information on the blockchain, users can reproduce the data workflow with precision, enhancing the reproducibility and reliability of scientific research outcomes.

As future development aimed at improving our system, we envision the deployment of a Blockchain as a Service (BaaS) platform on the private INFN-Cloud [18] infrastructure, providing users with customizable options for blockchain network configuration.

Thanks to this new feature, users will enhance their experience by having the flexibility to specify features such as the number of peers, channel structure, smart contract deployment, and consensus algorithm selection tailored to their specific use cases.

References

[1] INFN home page https://home.infn.it/en/

[2] ICSC home page https://www.supercomputing-icsc.it/en/icsc-home/

[3] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", 2008, https://bitcoin.org/bitcoin.pdf

[4] S. King and S. Nadal, "PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake", 2012, https://peercoin.net/assets/paper/peercoin-paper.pdf.

[5] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance", in *Proc. of the Third Symposium on Operating Systems Design and Implementation*, New Orleans, LA, USA, 1999, pp. 173–186.

[6] V. Buterin, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform", 2013, https://ethereum.org/en/whitepaper/.

[7] Hyperledger Fabric, "Hyperledger Fabric Documentation", https://hyperledger-fabric.readthedocs.io/.

[8] Go Programming Language, "Go Documentation", 2023, https://golang.org/doc/

[9] ECMA International, "ECMAScript® 2023 Language Specification", 2023 https://www.ecma-international.org/publications-and-standards/standards/ecma-262/

[10] Oracle Corporation, "Java Platform, Standard Edition & Java Development Kit Version Documentation",https://docs.oracle.com/en/java/javase/

[11] Flask, "Flask Documentation", Pallets Projects,https://flask.palletsprojects.com/.

[12] Indigo-IAM, https://indigo-iam.github.io/v/current/

[13] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore, "OpenID Connect Core 1.0", February 2014, https://openid.net/specs/openid-connect-core-1_0.html

[14] D. Hardt, "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, https://www.rfc-editor.org/rfc/rfc6749.html

[15] MongoDB, Inc., "MongoDB Documentation", 2023, https://docs.mongodb.com/.

[16] D. Ongaro and J. Ousterhout, "In Search of an Understandable Consensus Algorithm (Extended Version)", *USENIX Annual Technical Conference*, Philadelphia, PA, 2014, pp. 305-319.

[17] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, May 2015. https://www.rfc-editor.org/rfc/rfc7519.txt.

[18] INFN-Cloud home page https://www.cloud.infn.it/