Verifiable Querying Framework for Multi-Blockchain Applications

Stanly Wilson*‡, Kwabena Adu-Duodu*, Yinhao Li*, Ringo Sham*, Ellis Solaiman*, Omer Rana†, Rajiv Ranjan*
*Newcastle University, UK
†Cardiff University, UK
‡St. Vincent Pallotti College of Engineering & Technology, Nagpur, India

***Abstract*—Efficient and secure data retrieval from multiple blockchain networks at the same time remains a critical challenge. We propose a novel framework that supports authenticated data retrieval, ensuring integrity of the data to support both users and regulatory bodies. Our framework provides a scalable approach for information verification across diverse blockchain systems, enabling metadata from various systems to be combined to enable parallel querying across multiple blockchains.**

***Index Terms*—Blockchain, Metadata, Query**

I. Introduction

Blockchain is a distributed ledger technology (DLT) that is immutable, tamper-resistant and provides a decentralised store of chronologically recorded data. Unlike a traditional database, blockchain does not support all the CRUD operations (Create, Read, Update, Delete) due to its immutability. Hence, the update is just another insertion, and the delete operation is unavailable.

The transparent and immutable nature of a blockchain ensures that every transaction recorded is visible to all authorised participants. Retrieving information from multiple blockchains can be challenging due to the diverse nature of blockchain platforms and the widespread use of private blockchains. Interoperability issues arise as different platforms often use distinct data structures and consensus mechanisms, making it difficult to retrieve data seamlessly. The absence of standardized query interfaces and varying smart contract languages further complicates the process. Traditional blockchains only store data on a single topic, and there are no requirements for *join* operations. This is not the case while querying on multiple blockchains where a common field is required to perform the *join* operations [1]. The authors [2] had proposed a taxonomy for data management on the blockchain and this work uses concepts and ideas from it. This paper provides the following contributions: (i) a framework to verify information from multiple blockchains and which can scale to accommodate more entities; (ii) enabling users/ regulatory bodies to gather raw data using authenticated data structures (ADS) and execute authenticated range queries; (iii) enable data verification from multiple blockchains and perform authentication processes. The rest of the paper is set as follows. Section II discusses a few related works. Section III details the design of the framework. Section IV analyses the query process and integrity features. Section V provides framework evaluation. Section VI concludes the work.

II. Related Works

The paper [1] surveys query processing in blockchain-based systems and provides an overview of existing methods. The paper [3] examines querying mechanisms from three critical perspectives: efficiency, verifiability and security. For each of these perspectives, the paper further categorizes and explains the strengths and weaknesses of each approach. Additionally, it provides several use cases to illustrate the challenges that blockchain queries must overcome. The paper [4] compares searching or querying data in the blockchain. Existing platforms only allow queries with block hash or transaction hash. Extending querying that decouples the data available on each transaction, storing them on a secondary database (MongoDB), and queries them using traditional SQL is described. The framework (EtherQL) used the data from the Ethereum mainnet – claiming to be the first to make this querying possible. The paper [5] proposes FalconDB – based on the blockchain and uses ADS as a way to store and query data with multiple parties or nodes. It aimed to provide a collaborative platform where the nodes could work and share data without worrying about the presence of any malicious nodes. It uses ADS to store the data, and the digest generated from it is stored on the blockchain. The features of ADS coupled with blockchain enables moving storage to the server, and lightweight clients storing hashes. FalconDB could execute various queries and can provide provenance. Though FalconDB requires more storage on the server-side (to capture historical data), the overall performance is improved.

The paper [6] emphasises the role of ADS as an efficient way to store data but points out that the gas expenses for such operations are high. They propose a framework called Gas Efficient Merkle Merge Tree (GEM^2 Tree) that optimises the gas expenses by making bulk transactions rather than individual insertions at the cost of more computations required. The paper [7] identifies that blockchain has weak semantics and insufficient operations to support authentic queries. Hence they propose a model that combines on-chain and off-chain data for queries. They add relational semantic features to the data and provide an interface for executing these queries – the framework includes general information on-chain while the

private/sensitive information is off-chain. The on-chain data, based on the content, use indices for faster querying. SQL-like commands are used to create new tables, insert transactions and query the data using select commands. All these works are based on a single blockchain under consideration, and our use case must deal with multiple blockchains and organisations.

III. Verifiable Query Framework

Blockchain technology promises transparency and security, but accessing and querying distributed data is complex. Each blockchain operates as an independent entity with unique data structures, making it challenging to execute coherent queries across multiple platforms. Private blockchains restrict access to a defined set of participants, which complicates the querying process. These limitations hinder the free flow of information and demand innovative solutions for secure and authenticated data retrieval. There is a need for a verifiable querying framework, e.g. to allow regulatory bodies and users to execute authenticated queries. This will help in building trust and scalability in blockchain systems.

Consider a scenario with several organisations, with each organisation containing several entities: data owners, validators (blockchain is part of it), cloud ADS (CADS), query verifiers, query miners and query layer – as illustrated in Figure 1.

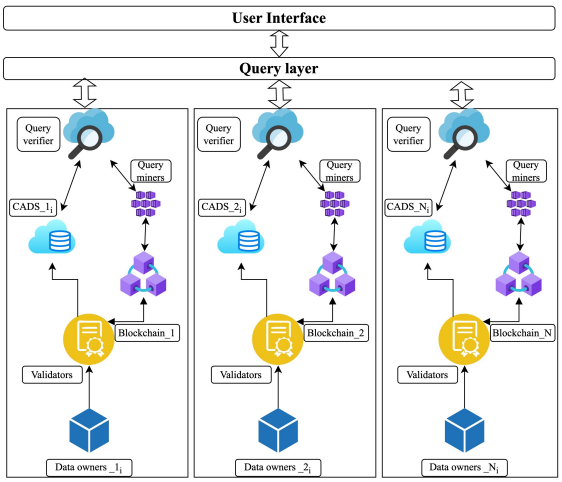

Fig. 1. System Architecture

Data Owners: decide what data should be made available through blockchain. They communicate directly and only to validators. In our scenario, data owners are the pharmaceutical, logistics and retail pharmacy companies who want to make some of their data available for public access.

Validators: are the entities that receive data from data owners and then push it to the blockchain. In our case, validators use the proof of authority (PoA) consensus algorithm, initially part of permissioned Ethereum. PoA requires fewer message exchanges leading to increased scalability. PoA is a reputation-based consensus protocol which can tolerate up to 50% malicious nodes. Proof of Work (PoW) consensus stakes its computational power, and Proof of Stake (PoS) consensus stakes coins in the consensus process. In contrast, PoA stakes its reputation, i.e. PoA consensus accepts node identity as a means to secure its blockchain. In PoA, a few validators are arbitrarily selected as trustworthy entities to verify a transaction. Since PoA relies on a few validators for consensus, the process is highly scalable [8]. Upon receiving the data d, the validators generate hash $h(d)$ and store it on the blockchain. After storing the data it would retrieve the following information from the blockchain: transaction data (TD), transaction hash (TH), block height (BN) and block hash (BH). The actual data d, and $h(d)$ are stored in the local database of the validator.

Cloud ADS (CADS): ADS have been used to improve the efficiency of an application by reducing the data retrieval process on the source of data [9]. ADS has been used on blockchain [6], [7], [10] and [5] provides a detailed description of the construction of ADS for a collaborative database platform. By its design, ADS supports clients to query and verify data (stored in the database) and provides proof of the data's correctness. Clients can check data integrity by retrieving its proof from the source (in our case, the blockchain). CADS can also return results for a collection of queries Q performed on its database D. CADS stores information sent only by the validators, and only verifiers can retrieve that information from the CADS. The authentication function and the security features of the CADS will be discussed in section IV.

Query Verifiers: do not store data but can query data from CADS and retrieve its proof from the blockchain. Upon receiving a query, the verifiers sends the request to CADS, which returns the results, transaction details and blockchain proofs (if requested).

Query Miners: Validators write data to the blockchain and not retrieve it during the query process. Query miners read transaction details from the blockchain during the query process. Hence, query verifiers send transaction details from CADS to query miners, who retrieve verification objects from the blockchain and return them to query verifiers. Query miners would communicate only with the query verifiers.

Query Layer: is responsible for collecting and processing query requests. Every organisation must send its metadata and query structure to the query layer, which has a metadata engine instructing where to search for data. The metadata engine helps the query layer to execute parallel searches across multiple systems, optimises the query process and aggregates the results. In some situations, organisations can change their schema; they can update the metadata with the query layer, ensuring that the query process returns the requested details. The metadata engine plays a crucial role in making the query layer scalable – i.e. to add additional organisations dynamically and manages the changes in schema from these organisations. Before updating the metadata, it validates the data to avoid errors. Figure 2 shows the query layer process.

IV. Query Process and Integrity Analysis

Metadata guides and orchestrates the query process by mapping attributes to their corresponding locations. The Meta-

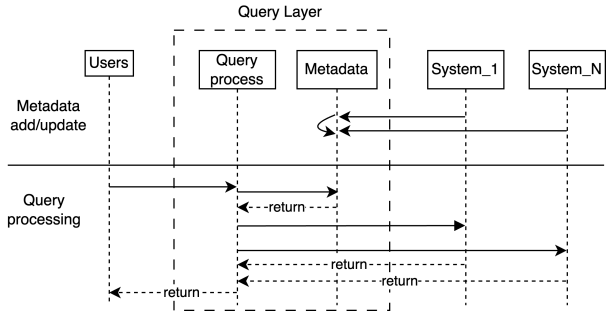

Fig. 2. Query Layer Process

data engine manages dynamic metadata changes and directs the query process to send requests to the correct locations. Upon receiving the metadata, it checks whether the received metadata has the correct structure. A query can either be a simple verification or a blockchain verification. Although the data retrieval process does not change the blockchain state, it still requires a gas fee to incentivize miners. Hence, separating the blockchain verification from simple verification (which returns results completely from CADS) allows the users to get the information from the systems without paying the gas fee. Figure 3 shows the sequence diagram illustrating how the query gets executed and retrieves information in a single system/ organisation.

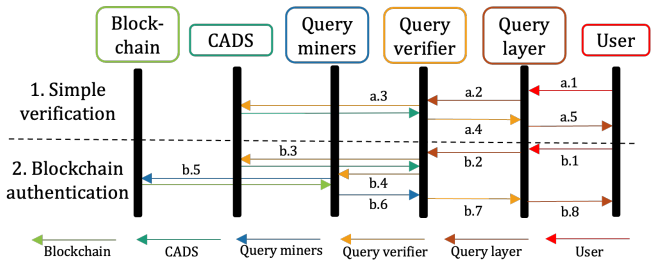

Fig. 3. Verification Process

The verification process uses features of CADS in providing fast data access and verifiability. In this process, a request from the user (a.1) reaches the query layer, which uses the metadata engine to send the query to the respective system(s) (a.2). The first entry point is to the query verifier, which diverts the query to CADS and gets its results (a.3). The query verifier then returns the results to the query layer (a.4) and then to the user (a.5). In the blockchain verification process, the request from the user (b.1) reaches the query layer, which, using the metadata engine to send the query to the respective system(s) (b.2), which diverts the query to CADS (b.3). The query verifier sends the verification object to the query miner (b.4), which retrieves the corresponding verification object from the blockchain (b.5) and returns it to the query verifier (b.6). The query verifier then compares and aggregates the results and sends them to the query layer (b.7). Finally, the query layer provides the results to the user (b.8). The initial processes in both simple and blockchain verification are the same. To reduce the complexity of the overall process, a flag is set to distinguish between the two processes. Depending on the condition, the system process query differently.

During the query process, verifiers obtain information from CADS, and it is important to ensure its integrity. By design, only read and write operations are permitted on CADS. The write permissions are only with validators and since the validators are trustworthy, the data they write on CADS will be reliable. The verifier nodes can only read the data from CADS and no other operations are possible for them. In practical deployment, there can be multiple CADS that are distributed to reduce latency and improve load balancing. The aspect of immutability is brought into CADS by removing the update or delete operations.

V. Framework Evaluation

We evalue the effectiveness of our prototype framework in this section. We have used a pharmaceutical supply chain scenario with 4 organizations: a manufacturing company, a logistic company and two pharmaceutical retail chains. Each entity manages its ADS, blockchains, validators and verifiers. Users can connect to the query layer to retrieve data, e.g. the complete product history. One of the assumptions is that the organizations have standardised naming conventions for their databases and provide the correct schema to the metadata engine. Every entity in the prototype currently implements asymmetric key cryptography and assumes that the keys are shared between the entities in a safe manner. The prototype uses 4 servers symbolizing 4 different organizations. Each server is hosted on an Intel Xeon E3-1220, having 16GB RAM running Ubuntu 18.04. Each organization manages their blockchain built on Ethereum and the ADS is developed using the MongoDB database. The prototype uses a pub-sub mechanism to communicate between the entities using symmetric and asymmetric key cryptography.

A. Performance Evaluation

We consider requesting several records from 4 organisations varying between 10,000 to 40,000 and assessing their latency. The evaluation is performed for both simple verification and blockchain verification. Experimental results were taken multiple times, and the averaged values are considered. In the case of simple verification, the requests returns the desired results within 0 to 3 seconds. Here, the records are returned only from ADS, not involving the blockchain. As the number of organisations increases from 1 to 4, the latency increases. This occurs due to the processing time taken to receive and process the information from multiple entities. Though the increase in the number of records does not affect latency significantly, the increase in the number of organisations increases latency as seen in figure 4.

When similar evaluations for blockchain verification are performed, the latency increases significantly. As discussed earlier, this process involves gathering information from ADS and the underlying blockchain. Blockchain processes contribute most significantly to the data retrieval process. As seen

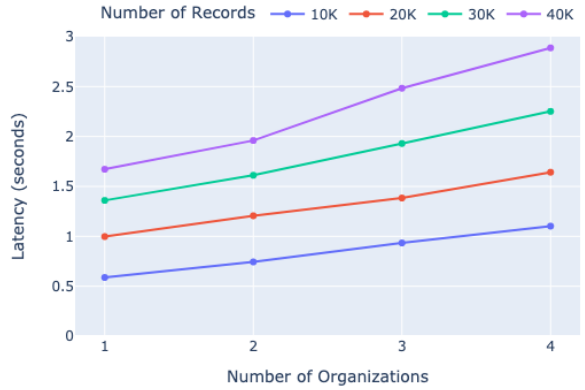

Fig. 4. Simple Verification Latency

in the figure 5, the majority of the total time required to retrieve information comes from the retrieval processes in the blockchain. It can also be noticed that the time required for retrieving information from the blockchain remains constant and is not affected by the increasing number of organisations.

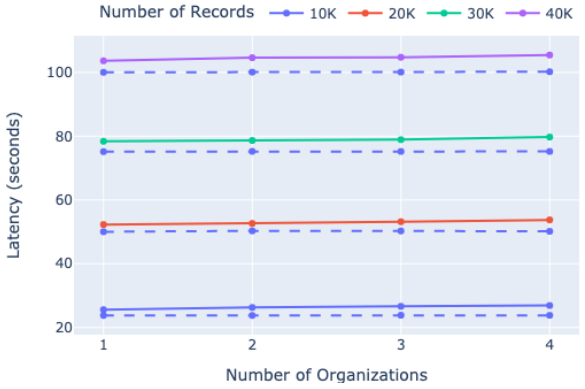

Fig. 5. Blockchain Verification Latency

B. Security and Privacy Evaluation

The framework we propose incorporates strong security and privacy measures by utilising a hybrid encryption model that combines the benefits of both asymmetric and symmetric encryption methods. Communication between each entity uses an asymmetric mechanism. Each time the information is sent, a randomly generated symmetric key is used to encrypt the data and is shared with the end user using the asymmetric key. Symmetric encryption is a fast and efficient way to encrypt and decrypt large amounts of data. It works by creating a new key for each transmission, which means that if a key is compromised, only the data that was encrypted with that specific key is at risk. This greatly reduces the amount of data that can be exposed in a security breach. The use of asymmetric encryption allows for the safe transfer of symmetric keys between parties and the end user. This approach guarantees that only the intended recipient, who has the appropriate private key, can decode the symmetric key that accompanies the encrypted data. This encryption layer effectively addresses the issue of key distribution that is present in symmetric systems, creating a secure means for exchanging keys without the risk of interception. This approach with two layers ensures that the information exchanged between entities is kept confidential and preserves privacy.

VI. Conclusion

A novel framework capable of retrieving information from multiple blockchains is proposed. The framework makes use of a robust metadata processing engine with secure querying methods. Robust data retrieval mechanisms are crucial as blockchain technology advances. Our work supports users and regulatory bodies by developing a resilient and adaptable blockchain ecosystem. Our metadata-based framework enables scalability, accommodating a broader range of entities and ensuring relevance in the evolving blockchain landscape. In the future, we plan to deploy our framework for real-world applications, evaluate different economic models, and implement a similar methodology in multilayer blockchains.

VII. Acknowledgements

Supported in part by the Engineering and Physical Sciences Research Council "Digital Economy" programme: EP/V042521/1 and EP/V042017/1.

References

[1] D. Przytarski, C. Stach, C. Gritti, and B. Mitschang, "Query processing in blockchain systems: Current state and future challenges," *Future Internet*, vol. 14, no. 1, 2021.

[2] S. Wilson, K. Adu-Duodu, Y. Li, E. Solaiman, O. Rana, S. Dustdar, and R. Ranjan, "Data management challenges in blockchain-based applications," *IEEE Internet Computing*, vol. 28, no. 1, pp. 70–80, 2024.

[3] Q. Zhang, Y. He, R. Lai, Z. Hou, and G. Zhao, "A survey on the efficiency, reliability, and security of data query in blockchain systems," *Future Generation Computer Systems*, vol. 145, pp. 303–320, 2023.

[4] Y. Li, K. Zheng, Y. Yan, Q. Liu, and X. Zhou, *EtherQL: A Query Layer for Blockchain System*, ser. Lecture Notes in Computer Science. Cham: Springer, 2017, vol. 10178, pp. 556–567.

[5] Y. Peng, M. Du, F. Li, R. Cheng, and D. Song, "Falcondb: Blockchain-based collaborative database," in *In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. ACM, 2020, pp. 637–652.

[6] C. Zhang, C. Xu, J. Xu, Y. Tang, and B. Choi, "Gem$\hat{2}$-tree: A gas-efficient structure for authenticated range queries in blockchain," in *IEEE 35th International Conference on Data Engineering (ICDE)*, 2019, pp. 842–853.

[7] Y. Zhu, Z. Zhang, C. Jin, A. Zhou, and Y. Yan, "Sebdb: Semantics empowered blockchain database," in *IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 1820–1831.

[8] P. B. Honnavalli, A. S. Cholin, A. Pai, A. D. Anekal, and A. D. Anekal, "A study on recent trends of consensus algorithms for private blockchain network," in *Blockchain and Applications*. Cham: Springer, 2020, pp. 31–41.

[9] R. Tamassia, *Authenticated Data Structures*. Berlin, Heidelberg: Springer, 2003, vol. 2832, pp. 2–5.

[10] H. Wu, Z. Peng, S. Guo, Y. Yang, and B. Xiao, "Vql: Efficient and verifiable cloud query services for blockchain systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1393–1406, 2022.