Blockchain PG: Enabling Authenticated Query and Trace Query in Database

Qingxing Guo, Sijia Deng, Lei Cai, Yanchao Zhu, Zhao Zhang(✉), and Cheqing Jin

School of Data Science and Engineering, East China Normal University, Shanghai, China
{qingxingguo,sjdeng,leicai,yczhu}@stu.ecnu.edu.cn,
{zhzhang,cqjin}@dase.ecnu.edu.cn

Abstract. Blockchain comes under the spotlight for successfully implementing a tamper-resistant ledger among multiple untrusted participants. The widespread adoption of blockchain in data-intensive applications has led to the demand for querying data stored in blockchain databases. However, compared to traditional databases, current blockchain systems cannot offer efficient queries. Moreover, migrating the original business supported by traditional databases to blockchain systems takes a long time and costs a lot. Motivated by this, we design and implement Blockchain PG, a novel data management system built on a legacy system. The system architecture applies to most databases system. By establishing a trusted relationship between the database and the client, Blockchain PG not only guarantees that the existing legacy system will not be greatly affected, but also achieves the data integrity and correctness by authenticated query, and the traceability of data by trace query.

Keywords: Blockchain · Authenticated query · MBtree · Trace query.

1 Introduction

The centralized database might be compromised and authorized users can abuse privileges to disguise the identity of others to maliciously operate data, making data security becomes a serious problem [6]. Recently, the appearance of blockchain offers a promising direction for protecting data. As a distributed append-only ledger maintained by multiple parties, Blockchain provides reliable data storage service in a potentially hostile environment [1].

When blockchain is widely used by many applications in various fields, query requests are growing in number and complexity [8]. So far many efforts have been devoted to improving data management of blockchain [2,7,9]. By integrating the blockchain technique and database characteristics, the blockchain database has been optimized to support rich queries. For example, SEBDB [10] achieves the APIs interface of SQL-like language by adding relational semantics

X. Wang et al. (Eds.): APWeb-WAIM 2020, LNCS 12318, pp. 529–534, 2020.
https://doi.org/10.1007/978-3-030-60290-1_41

to blockchain systems. However, when the business involves more complex functions that blockchain systems do not support, like aggregate queries, it will be tricky to migrate them from a traditional database to blockchain systems without changing the original functions. Besides, current blockchain platforms run in a distributed environment, resulting in low efficiency in submitting transactions through a consensus protocol.

This paper proposes Blockchain PG modified based on PostgreSQL. The system architecture is also applicable to other databases. Firstly, we build Authenticated Data Structure(ADS)[5] on tuples to ensure that client can verify the integrity and correctness of data through authenticated query. Then, the history table is designed to track user operation and table rows are chained through cryptographic hashes to ensure tamper-evident data. The history table empowers trace queries to detect illegal behaviors of authorized users. Finally, establishing a trusted relationship between the system and the client without consensus protocol can greatly improve query performance.

2 System Overview

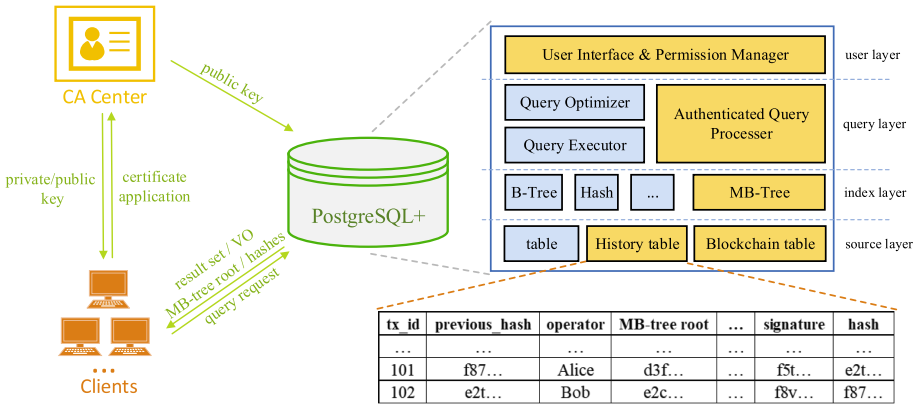


Fig. 1. System architecture.

As shown in Fig. 1, Blockchain PG consists of three parts, CA center, client, and PostgreSQL+. The CA center is responsible for distributing certificates that can prove the client's identity. PostgreSQL+ is a database system divided into four layers: (i) user layer works to process user request and verify the user's identity with the public key; (ii) query layer generates a plan for authenticated query or normal query; (iii) index layer performs as data retrieval, and MB-tree [4] is built as an authenticated index structure at this layer; and (iv) source layer works to store table files, including blockchain tables and history tables. Compared with an ordinary table, the blockchain table is a new type with an MB-tree index.

The history table is insert-only and highly tamper-resistant. Table rows are chained by cryptographic hashes based on previous row hash and current row data.

2.1 Identity Management

CA Center provides stronger database security against user fraud. It will generate a pair of public and private keys upon receipt of the client's certificate request, and then send the private key to the client and broadcast the public key to other clients and PostgreSQL+. Message signed by the client's private key can be decrypted and its owner will be verified by the corresponding public key stored on others [3].

2.2 Maintaince of Verification Information

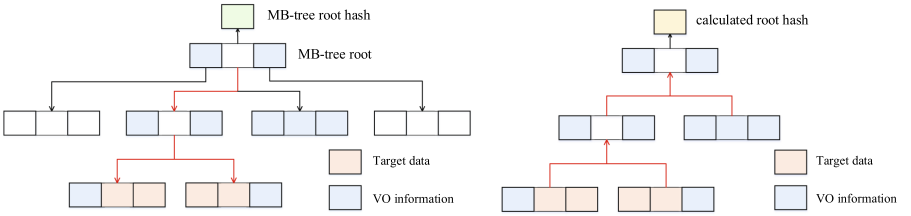
If a client successfully connects to PostgreSQL+, it will synchronize the MB-tree root and hashes taken from the history table at specific intervals, including the latest hash value, to the client as initial verification information.

Once the client initiates update requests along with the message signature through the user's private key, the MB-tree index in PostgreSQL+ will be updated and generate the newest hash root. Next, PostgreSQL+ inserts the update operation into the history table and calculates a new row hash. Finally, the newest MB-tree root and row hash are synchronized to all clients for a basis for the client to verify the query result. Expired hashes left on clients are not discarded completely, but stored by clients at certain intervals. For example, the hash column in the history table of Postgresql+ is (h0, h1, h2, ..., h102), while the client stores (h0, h10, h20, ..., h100, h101, h102).

2.3 Queries

As a centralized database might be untrusted, we proposed authenticated query and trace query in Blockchain PG for clients to verify the correctness and completeness of the data and tracking the illegal behavior of authorized users. Now we detail authenticated query and trace query procedures.

Authenticated Query. When Blockchain PG receives an authenticated query request, it generates an authenticated query plan that allows the executor to obtain tuples of the blockchain table through the MB-tree index in the query layer. As suggested in Fig. 2(a), Query executor accesses nodes in the MB-tree from root to leaf and appends hashes of sibling nodes to verification objects(known as VO), and then sends VO and query result to the client. The client calculates a new MB-tree root hash from VO and result set as demonstrated in Fig. 2(b). We can judge whether the result set is correct and complete by comparing the calculated hash value with the latest MB-tree root saved locally on the client.



(a) Range query in MB-Tree. (b) Verify the results set through VO.

Fig. 2. Query and verification of Authenticated query.

Trace Query. Trace query provides an approach to track user behavior. Since the client stores the hash column of the history table at intervals, our query strategy does not need to return the whole data of the history table to determine whether the table has been changed. If the client wants to know data changes of blockchain tables in a certain period, it first finds two boundary hash values containing the query target in the local hash set, and then query all data in the boundary interval from the history table of PotsgreSQL+. After the client receives the query result, it will hash the data and reorganize them into a hash chain. We know whether the log record has been modified by comparing the two ends of the hash chain with the previous boundary hashes. Moreover, the signature bound to the user's private key in the history table provides identity fraud protection.

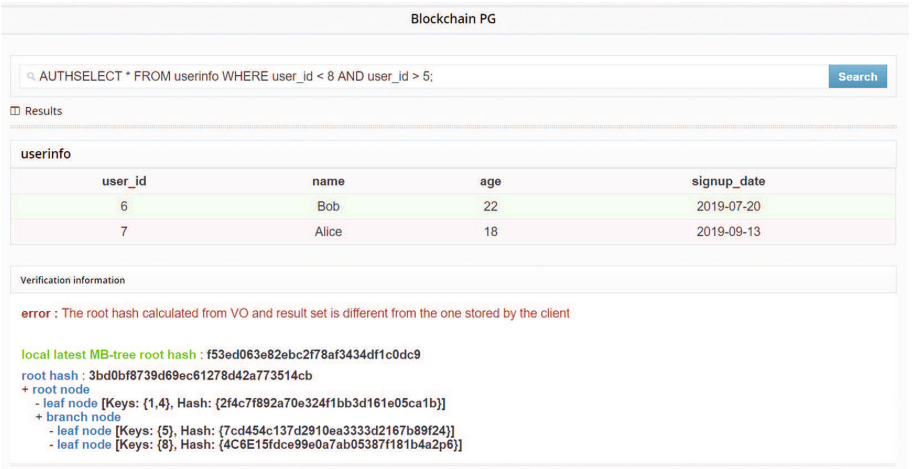


Fig. 3. The interface of Authenticated Query. (Color figure online)

3 Demonstration and Evaluation

Blockchain PG is implemented on the basis of PostgreSQL which is an excellent open-source database. The system architecture is also applicable to other databases. The whole system runs on Ubuntu 16.04 OS. The client protocol has been modified to accommodate authenticated and trace queries. Figure 3 and Fig. 4 show the response results when the client sends authenticated query and trace query requests, respectively. We create a userinfo table containing name, age, and other fields in PostgreSQL+, and implement a data generator which supports uniform and Gaussian distribution of data to simulate the real scenario. In Fig. 3, The VO structure, including leaf node and branch node, marked in blue font is presented in the form of a tree. We can calculate a root hash value through VO and the result set of the query, and then compare the hash value with local root hash to check whether the query result set is correct and complete. As the data in the blockchain table is maliciously modified by other authorized user, the client find that the data is wrong when verifying the query set. In addition, we can track user behavior through trace queries illustrated in Fig. 4. Since target records that the client wants to find is in the hash interval marked in red font, the client only needs to look up the records in the hash interval to prove that the records have not been modified, thus correctly tracks the user's behavior.

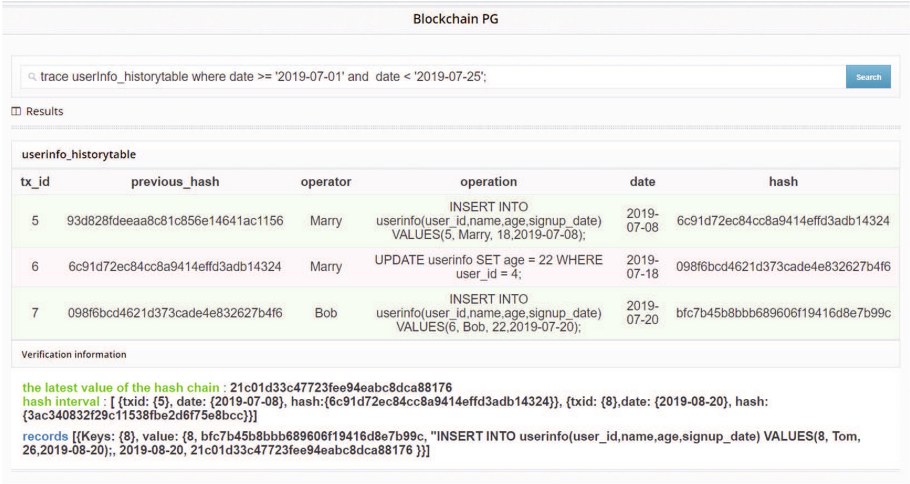


Fig. 4. The interface of Trace Query.

Acknowledgments. This research is supported by in part by National Science Foundation of China under grant number U1811264, U1911203, 61972152 and 61532021.

References

1. Chitti, P., Murkin, J., Chitchyan, R.: Data management: relational vs blockchain databases. In: Advanced Information Systems Engineering Workshops - CAiSE 2019 International Workshops, pp. 189–200 (2019)
2. El-Hindi, M., Binnig, C., Arasu, A., Kossmann, D., Ramamurthy, R.: Blockchaindb - A shared database on blockchains. In: Proceedings VLDB Endowment, pp. 1597–1609 (2019)
3. Hyla, T., Pejas, J.: Long-term verification of signatures based on a blockchain. Comput. Electr. Eng. **81**, 106523 (2020)
4. Li, F., Hadjieleftheriou, M., Kollios, G., Reyzin, L.: Dynamic authenticated index structures for outsourced databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 121–132. ACM (2006)
5. Martel, C.U., Nuckolls, G., Devanbu, P.T., Gertz, M., Kwong, A., Stubblebine, S.G.: A general model for authenticated data structures. Algorithmica **39**(1), 21–41 (2004)
6. Mathew, S., Petropoulos, M., Ngo, H.Q., Upadhyaya, S.J.: A data-centric approach to insider attack detection in database systems. In: RAID, pp. 382–401 (2010)
7. Ruan, P., Chen, G., Dinh, A., Lin, Q., Ooi, B.C., Zhang, M.: Fine-grained, secure and efficient data provenance for blockchain. In: Proceedings VLDB Endowment, pp. 975–988 (2019)
8. Xu, C., Zhang, C., Xu, J.: vchain: enabling verifiable boolean range queries over blockchain databases. In: Proceedings of the 2019 International Conference on Management of Data, pp. 141–158. ACM (2019)
9. Zhang, C., Xu, C., Xu, J., Tang, Y., Choi, B.: Gem$\hat{}$2-tree: a gas-efficient structure for authenticated range queries in blockchain. In: ICDE, pp. 842–853 (2019)
10. Zhu, Y., Zhang, Z., Jin, C., et al.: SEBDB: semantics empowered blockchain database, pp. 1820–1831. IEEE (2019)