Zero-knowledge data validation method based on the homomorphic hash function in the distributed decentralized storage platform

Stanislav Bogatyrev, Anatoly Bogatyrev, Sergei Liubich, and Fabian Wahle

JSC "NEO Saint Petersburg Competence Center", St. Petersburg, Russia
{info,stanislav,anatoly,sergei,fabian}@nspcc.ru
https://nspcc.ru

Abstract. The design of the Distributed Decentralized Storage Platform (DDSP) requires to develop an efficient data validation method taking into account the network scalability issue, possibility to check data without knowledge of content and privacy protection of users data. DDSP is aimed at ensuring the availability and integrity of data. The aim of this research is to develop a zero-knowledge data validation method for the decentralized storage system for minimizing data transferring to maintain the network scalability and minimizing computational cost on the side of a storage node to maintain a large number of parallel interactions, and on the side of a validating node.
The integrity guarantee is achieved due to the proposed data validation method based on the homomorphic hash function which allows to verify the integrity of data on a storage node without transferring real data to a validating party over the network.
The computational complexity of validations depends linearly on the size of validated data and is well suited for parallelization.

Keywords: Zero-knowledge · Data validation · Zero-knowledge data validation · Homomorphic hash · Distributed · Decentralized · Storage platform · Object storage · Storage.

1 Introduction

Nowadays, building an efficient, reliable and scalable decentralized data storage architecture is an actual problem in both corporate and academic communities since decentralization requires new approaches that adapt existing models.

In the decentralized storage system, where users no longer physically own the storage of their data, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted [1].

A group of works [1–4] have been done focusing on attempt to solve a remote data validation task in cloud storage. These methods can be classified as Proof of Data Possession(PDP) and Proof of Retrievability(PoR).

The PDP scheme initially has been presented by Ateniese [2, 3]. Related protocols detect a large amount of corruption in outsourced data. However, the schemes do not support the possibility to allow an external party to verify the correctness of remotely stored data without knowledge of data content, what is needed for decentralized storage systems.

An efficient PDP method with privacy protection of users data from external auditors on a client and server sides is needed for decentralized storage. From the perspective of data privacy protection, this drawback greatly affects the security of the protocols [5].

The design of the Distributed Decentralized Storage Platform (DDSP) requires to develop an efficient data validation method [7] taking into account the network scalability issue, possibility to check data without knowledge of content and privacy protection of users data.

This paper proposes a zero-knowledge data validation method for the decentralized storage system for minimizing data transferring to maintain the network scalability and minimizing computational cost on the side of a storage node to maintain a large number of parallel interactions, and on the side of a validating node [8]. The integrity guarantee is achieved due to the proposed data validation method based on the homomorphic hash function [9] which allows to verify the integrity of data on a storage node without transferring real data to a validating party over the network.

2 Data Validation Method

Requirements for the data validation [7] method for DDSP:

– the ability to publicly verify stored data without an actual object ownership and the knowledge of an object content (zero-knowledge proof);
– validation needs to ensure that a storage node's response cannot be saved and repeated to counteract nodes-malefactors (method has to be developed in accordance with an arbitrary task of validation);
– minimization of a computational cost on the side of a storage node to ensure a large number of parallel interactions and on the side of a verifying node;
– minimization of a network load to maintain a network scalability.

In order to fulfill the requirements, the data validation method based on homomorphic hashing [9] is proposed.

A hash function has to have the following properties:

– it should be easily computable;
– it should be computationally difficult to find collisions [6].

Homomorphic hash is a hash function that can compute a hash of a composite block from hashes of individual blocks.

The computational complexity of validations depends linearly on the size of validated data and is well suited for parallelization.

In the proposed method, the challenge-response model is applied. The challenge can be formed as a collective game for all storage nodes that contain validated data. In the simplest case, a request for an arbitrary set of hashes from the data being verified is considered. The obtained hashes have to produce the expected hash according to the rule of homomorphism. Detailed description of the challenge and the analysis of resistance to attacks are the subjects of future research.

Validation of each individual object is a computationally expensive task for a decentralized system with an unlimited number of users and the amount of stored data. A storage group is introduced to reduce a computational cost and the amount of stored meta-information for validation.

2.1 Data Validation Complexity: Storage Group

The concept of a storage group is introduced to reduce a validation complexity dependence on the number of stored objects in the system. The storage group encapsulates a group of objects' identifiers and a set of homomorphic hashes required for data validation. The safety and accessibility of multiple objects in the network are achieved by the storage group validation without storing meta-information and conducting validation of each object. The concept of a container of objects is introduced. One container can have any number of storage groups. The storage group is an immutable structure, however, new storage groups can still be formed based on the existing ones (the merge operation) or the storage group can be removed (the objects will be deleted). A group of homomorphic hashes is generated for data validation. One hash is created from all the ordered data of all storage group objects (zero-level hash) and a fixed number of hashes is created from zones (first-level hashes). The zone‘s size is a multiple of the size of the object‘s block. A fixed and equal number of homomorphic hashes of each level is stored for all the groups in DDSP. As an example – retrieving data for the storage group, where the zero-level hash consists of six objects. The case, where it is required to store four first-level homomorphic hashes, is shown on Fig. 1 below. The actions to form the storage group are following:

- getting of a sorted list of homomorphic objects‘ hashes: $h(O_1)$ ... $h(O_6)$;
- getting of a single structure from ordered objects‘ data;
- dividing of the data structure into four zones $D_{1.1}, D_{1.2}, D_{1.3}, D_{1.4}$ in order to form homomorphic hashes;
- getting of first-level hashes $H_{1.1} = h(D_{1.1})$ ... $H_{1.4} = h(D_{1.4})$.

In the simplest case, a validating node selects a random first-level homomorphic hash during data validation and requests some number of second-level homomorphic hashes from the placement group nodes. This procedure is a collective game. Merging of the second-level homomorphic hashes, a validating node confirms the validation if the obtained hash corresponds to the first-level hash being stored.

3 Results

The proposed method allows to minimize the load on the network and validating nodes without transferring a real data over the network. The storage group

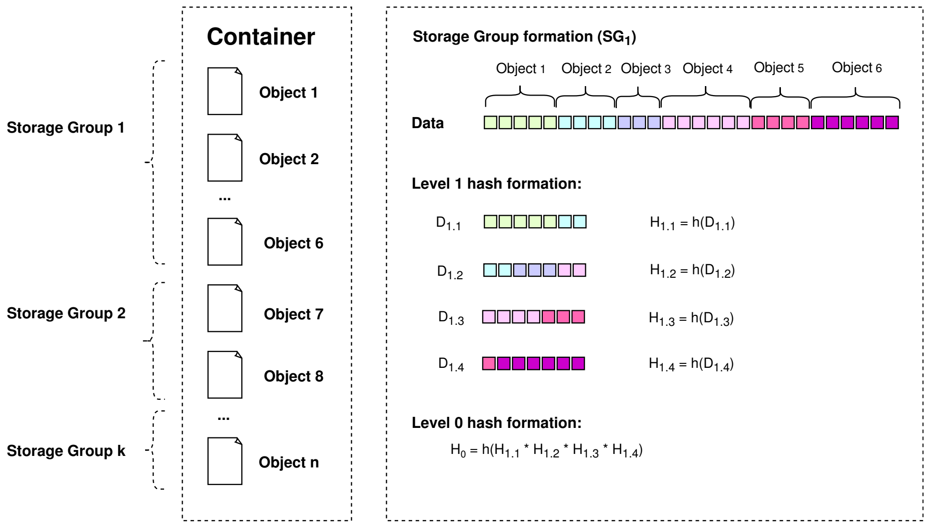

Fig. 1. Example of forming the storage group.

allows to keep a fixed amount of meta-information needed for validation process regardless of the size and the number of objects from the storage group. The comparison of the proposed method with the existing ones, a detailed challenge description and the analysis of resistance to malicious attacks are the subjects of further research.

References

1. Wang, S.H., Chen, D.W., Wang, Z.W., Chang, S.Q.: Public Auditing for Ensuring Cloud Data Storage Security With Zero Knowledge Privacy. In: IACR Cryptology ePrint Archive, Report 2012/365 (2012)
2. Ateniese, G., Burns, R., Curtmola, R.: Provable data possession at untrusted stores. In: Cryptology ePrint Archive, Report 2007/202 (2007)
3. Ateniese, G., Pietro, R.D., Mancini, L.V.: Scalable and efficient provable data possession [C]. In: Proceedings of the 4th international conference on security and privacy in Communication networks. Istanbul, Turkey: ACM, pp. 90-99 (2008)
4. Wang, Q., Wang, C., Li, J.: Enabling public verifiability and data dynamics for storage security in cloud computing[C]. In: Proc. of ESORICS'09, Saint Malo, France, Sep. 2009. Lecture Notes in Computer Science, vol. 5789/2009, pp. 355-370 (2009)
5. Shah, M.A., Baker, M., Mogul, J.C., Swaminathan, R.: Auditing to keep online storage services honest[C]. In: Proc. of HotOS'07, pp. 1-6 (2007)
6. Zemor, G.: Hash functions and graphs with large girths. In: EUROCRYPT 91 (1991) LNCS 547 Springer-Verlag, pp. 508-511 (1991)
7. Ateniese, G., Pietro, R.D., Mancini, L.V., Tsudik, G.: Scalable and Efficient Provable Data Possession. In: Proceedings of the 4th international conference on Security and privacy in communication networks (SecureComm '08) (2008)

8. Waters, B.: Compact Proofs of Retrievability. In Journal of Cryptology, vol. 26, no. 3, pp. 442–83 (2013)
9. Mullan, C., Tsaban, B.: SL2 Homomorphic Hash Functions: Worst Case to Average Case Reduction and Short Collision Search. In Journal Designs Codes and Cryptography, vol. 81, no. 1, pp. 83–107 (2016)