Hyperledger-based Data Provenance in Distributed Computing Environments*

A.P. Demichev[1], A.P. Kryukov[1], N.V. Prikhod'ko[2], J.Yu. Dubenskaya[1], E.Yu. Fedotova[1] and S.P. Polyakov[1]

[1]Skobeltsyn Institute of Nuclear Physics, Lomonosov Moscow State University, Moscow, Russia,
[2]Yaroslav-the-Wise Novgorod State University, Velikiy Novgorod, Russia

A new approach to managing provenance metadata and data access rights in a distributed environment is proposed. It is based on the integration of blockchain technology, smart contracts and provenance metadata driven data management. We also suggest a new method for delegation of rights from a user or service to another service within distributed computing systems. The implementation of the proposed approach, entitled ProvHL, is based on the permissioned blockchains and on the Hyperledger Fabric blockchain platform in conjunction with Hyperledger Composer.

Keywords: distributed storage, provenance metadata, blockchain, smart contracts, access rights, delegation of rights, Hyperledger Fabric.

1. Introduction

When implementing large-scale scientific or business projects related to storing and processing large amounts of data and involving participants from different administrative domains, the best solution may be that the project participants combine their local storage resources into a single distributed pool and, if necessary, rent additional cloud storage resources, possibly from several providers. Indeed, in existing centralized solutions, the main functions are performed by data centers that collect and store (possibly with subsequent processing) data from peripheral (user) nodes of the network. Therefore, in this case, the need to store big data in any scientific or production area leads to the necessity to build large and very expensive specialized data centers. At the initial stage of implementing a project, it is very problematic both to find sufficient funding for the establishment of such a center, and to estimate in advance the necessary storage capacity for a sufficiently long period of time. The approach based on the peer-to-peer (P2P) paradigm of storage networks (see the review [1] and references therein) is totally opposite to the completely centralized approach. In this case, data storage services are evenly distributed among all network participants, which provides a natural load balancing, the absence of bottlenecks and points of failure. Special mechanisms of coding, fragmentation and distribution of information over nodes can provide privacy and reliability of the system even in case of failure of some storage nodes. However, a significant problem with this approach is to ensure a stable pool of peers, that is storage resource providers, especially at the initial stage of development of such a network. In other words, before such a P2P-based storage can work stably, it will require significant technical, organizational and time costs from its organizers in the absence of a result guarantee, that is, a workable network with sufficient storage capacity. Therefore, in many cases of large-scale projects, the above-mentioned solution that is intermediate between the fully centralized and fully decentralized (P2P) solutions may be optimal. In science, examples of such large projects are the Large Hadron Collider (CERN, Geneva; https://home.cern/science/accelerators/large-hadron-collider) and experiments in astrophysics (see e.g., [2]).

In the case of this intermediate approach, the problem arises of combining all local storages and data in them into a unified storage system in a dynamically changing environment, as well

*This work was funded by the Russian Science Foundation (grant No. 18-11-00075).

as ensuring the implementation of mutual access policies to the data of the parties involved. This implies the existence of methods for decentralized management of data access rights in such a dynamically changing environment, ensuring consensus among the parties involved on the content and order of data operations and ensuring reliable, immutable recording of proven operations history, that is, provenance metadata. The latter are necessary for data storage and usage consistency, as well as for consideration and resolving possible conflicts among project participants or with the storage providers. In other words, it is necessary to provide tools to support the implementation of business processes of storing and exchanging data in a distributed environment and in the presence of administratively unrelated or loosely related organizations involved in joint projects, or simply sharing data under certain conditions.

In this work, we propose an approach to solving this problem based on the use of blockchain technology and smart contracts within the Hyperledger platform (https://www.hyperledger.org) [3]. The basic principles of operation, architecture and algorithms of the ProvHL system (Provenance HyperLedger) for managing provenance metadata and data access rights in distributed storages are developed. At present, a testbed has been created in SINP MSU, on which a preliminary version of the ProvHL prototype is deployed for elaboration of these principles and algorithms. The paper is organized as follows. In Section 2 we describe business processes of data storage and sharing in distributed environments. In Sections 3 and 4 the Hyperledger blockchain platform and distributed consensus algorithms are presented, respectively. Section 5 is devoted to presentation of the ProvHL system and Section 6 describes delegation of rights in the framework of this system. In Section 7 a short review of related works is given and, finally, Section 8 contains conclusions.

2. Business Processes of Data Storage and Sharing in Distributed Environments

The basic scenario of using the proposed system assumes that a virtual organization (VO) is formed for the joint implementation of a certain project. VO includes several real organizations which include, in turn, data providers, users and data handlers affiliated with them. It is assumed that the implementation of such a project requires the use of a distributed data storage. This distributed storage can be formed by renting multiple cloud storages, as well as integrating the own storage resources of the organizations that form the VO. Thus, the hardware and software basis of the business environment in this case is formed by a set of storages (possibly of different types, e.g., cloud storages, file servers, tape storages, etc.), each of which can be managed by its own data management system (DMS). For certainty, it is further assumed that the data is stored as files, i.e. the file is a unit of data. Generally speaking, several VOs can coexist; the storages with which they interact can form partially overlapping sets.

In such an environment, an immutable and distributed (as the environment itself) registry and a consensus on the order of data operations are needed to resolve possible conflicts between the VO/project participants related to the use of the data. In other words, to resolve possible conflicts, one needs an undistorted history of data use by VO members. Conflicts may be caused by priority issues upon obtaining results of data processing, use of results (including funding issues), etc. A simple example of the causes of such conflicts are cases of deliberate or accidental violation of the data access rights policy. Another example is when, based on shared resources, participants from different administrative domains obtain similar important results, and questions arise about how independently they were received and which of the participants obtained them first.

The state of a data unit (file) is determined by its provenance metadata (PMD), which consist of its global file identifier (ID) and its attributes, including: (1) local file name in a storage; (2) storage identifier; (3) creator identifier; (4) owner identifier; (5) source; (6) number of file downloads, etc. Here we mentioned only the most common possible attributes. The attribute

set chosen may depend on the needs of a particular project using the distributed storage. The full set of possible attributes is defined by the corresponding standard. In particular, in this work we rely on one of the most widely used PMD standards, namely, W3C PROV [4]. The set of values for all attributes of a file determines its current state. The state of the entire distributed storage system is determined by the set of files stored in it with their states at the moment.

It is supposed that the distributed storage works in the framework of immutable shared files semantics [5]. Basic operations (comprising of a set of transactions) can be of the following types: new file upload; file download; file copy within a storage; file deletion; file copy to another storage; file transfer to another storage. Each active transaction, and therefore operation, corresponds to an update of some state keys, for example, after the operation "file download" the values of the following keys change: "number of file downloads" and "users who downloaded the file". There are, of course, transactions for changing file attributes, for example, changing the owner of a file.

In addition to the task of recording the immutable history of working with data in a distributed storage environment, there exists the task of providing distributed management of access rights to data. For example, the owner of a data file (the user who created the data or the organization to which it belongs through its authorized representative) must be able to manage its access rights for other users. Another example is when a cloud storage service grants access to data stored on it only to users from organizations that have paid for this storage service.

3. Hyperledger Blockchain Platform

A natural solution for the establishment of a distributed immutable registry for the PMD records is the use of the blockchain technology. To implement this solution, it is convenient to use existing blockchain platforms. We require the platform to be used to have the following properties: working with permissioned blockchains; using smart contracts to manage transactions and organize business processes for sharing data and storage resources by project participants located in different administrative domains; availability of tools for managing access rights to certain actions with metadata; using a modular structure that allows one to use various algorithms to achieve consensus between the participants of business processes, depending on the needs of a specific distributed data computing system and its users; possibility of simultaneous independent work of several virtual organizations. Analysis of existing platforms (see, e.g., the reviews [6–8] and refs. therein) shows that the requirements above practically unambiguously distinguish the Hyperledger Fabric blockchain platform (HLF; www.hyperledger.org) [3] together with Hyperledger Composer (HLC; hyperledger.github.io/composer) as the best candidate (HLF&C-platform). The Hyperledger Composer is a set of tools for simplified use of the blockchain.

A smart contract along with the registry form the basis of a blockchain system. While the registry contains information about the current and historical state of a set of business objects, a smart contract determines the executable logic that generates new states to be added to the registry. Before parties of a business process can enter into interactions with each other, they must define a common set of contracts covering common terms, data, rules, concept definitions and processes. Taken together, these contracts define a business model that governs all interactions between transactional parties. A smart contract defines these rules between the parties in the form of executable code. In the framework of the HLF&C-platform the smart contracts are implemented under the name chaincode. In particular, access rights to data files are defined by the smart contracts in conjunction with special system acl-file ("acl" stands for "access control language").

To describe the business process within the framework of HLF&C-platform, a number of

concepts are used, the main ones are assets, participants, transactions and events. Assets are tangible or intellectual resources, services or property, records of which are kept in registries. Assets can represent almost anything in a business network, such as a house for sale, a listing for sale, a land registration certificate for that house, while insurance documents for that house can be assets in one or more business networks. Assets must have a unique identifier, but they can also contain any properties defined for them. In our case, the assets are data files; their properties (attributes) are provenance metadata, as defined in the previous Section 2.

Participants are members of the business network. They can own assets and make transaction requests. Like assets, the participants must have an ID and can have any other properties if necessary. The transaction is the mechanism of interaction of participants with assets. The definition of events is also established in the process of a business network construction, similarly to the assets or participants. According to these definitions, event messages can be sent by transaction processors to inform external software components of changes in the blockchain. Applications can subscribe to receive event information via the HLC API.

From a functional point of view, the nodes in the HLF network are divided into three types: clients make requests to execute transactions, participate in their processing, and broadcast transactions to ordering services; peers carry out the transaction processing workflow, validate them and manage the blockchain registry; ordering service nodes (OSN) establish the general order of all transactions in the blockchain based on the distributed consensus algorithm. Unlike public blockchain networks, which allow non-authenticated users to participate in their work, members of the HLF&C-network must be registered with Membership Service Provider (MSP), which, among other things, performs the functions of Certification Authority (CA). It should be emphasized that although the CA is a centralized service, it does not violate the decentralized nature of the HLF&C-network, distributed storage and PMD management system for two main reasons: (1) the CA is not a party to the business process, but only a trusted third-party that provides cryptographic material (digital certificates); although HLF&C provides software for the organization of its own CA for the deployed network, third-party public CAs can also be used; (2) the CA is not involved in the process of transaction processing and block formation, so it does not affect system performance and fault tolerance. In other words, the CA is outside the business network. Other MSP components, in particular those providing authentication and authorization of participants, are distributed across network nodes.

4. Consensus Algorithms

As it was mentioned in the Introduction, data management via PMD requires a method of ensuring consensus among participants in the business process about the content and order of transactions with data. Nowadays there exits a number of consensus algorithms that do not require resource-consuming and slow “proof-of-work” mechanism which is intrinsic for cryptocurrency blockchain networks [9].

One of the first and most well known consensus algorithms is the Paxos algorithm [10]. This algorithm is not designed to work in distributed systems with possible Byzantine errors. It is very difficult for understanding and implementing [11]. The Raft algorithm [11, 12] realizes the consensus by choosing a single leader, giving it full responsibility for managing the transaction recordings. The leader accepts requests from clients, copies them to other servers, and tells the rest of the servers when it is safe to use the entries in their replicated ledgers. The idea of having a special leader simplifies the management of the replicated ledger. If the leader for some reason stops working, the procedure for selecting a new leader begins. However, Raft is also not designed to work in distributed systems in which a Byzantine type of error is possible (malicious distortion of information by nodes).

The Practical Byzantine Fault Tolerance (PBFT) algorithm [13] was the first practical solution to achieve consensus in the face of Byzantine failures. It uses the concept of replicated

state machine, and nodes in a PBFT system are sequentially ordered with one node being the leader and others referred to as backup nodes. All nodes in the system communicate with one another with the goal being that all honest nodes will come to an agreement of the state of the system using a majority rule. This algorithm requires $3n + 1$ replicas to be able to tolerate n failing nodes. Communication between nodes has two functions: nodes must prove that messages came from a specific peer node, and they must verify that the message was not modified during transmission. This approach imposes a low overhead on the performance of the ordering service nodes (OSNs). However messaging overhead in the case of PBFT increases significantly as the number of OSNs increase. However it remains acceptable for a couple of dozens of OSNs (parties in a project using a distributed storage under the blockchain-based management) [13, 14]. Currently we consider PBFT as a most suatable distributed consensus algorithm since its properties match the system requirements and there exist exploratory studies for its implementation within the HLF&C platform [14]. In the future we plan to consider and test other BFT algorithms, such as SIEVE [15], XFT [16] and Hashgraph [17].

5. ProvHL System

This section outlines the basic principles of our ProvHL system for managing provenance metadata and access rights to data in distributed storages based on the HLF&C blockchain platform.

In general, two approaches are possible: (1) data management systems (DMS) manage data and use blockchain as a distributed log (data driven data management); (2) first, the metadata is written to the blockchain, and DMSs refer to the blockchain and performs the transactions recorded there (metadata driven data management). In the first case, the functionality of the blockchain system is very limited, it only provides a ledger which is resistant to malicious attempts to modify the history of data in distributed storage. HLF&C-platform enables one to implement the second approach, which in addition to simply maintaining the registry allows us to solve the problem of distributed data access management. Note that the term “metadata driven” is most often used in the context of ETL-technology (Extract, Transform, Load); we use it solely to designate a way to manage data by pre-writing metadata to the HLF.

For each data operation, at least two types of transaction records are made in the blockchain: one corresponds to the client request, and the second corresponds to the server response.The algorithms used are aimed at ensuring consistency between the state of the distributed storage and the entries in the blockchain. In particular, when the “new file upload” transaction is performed, the transaction on creating the new asset, that is the data file, with the “temporary” label is first recorded in the blockchain. And only after the actual upload of the file in the storage, DMS initiates a transaction removing the label “temporary” and turns the uploaded file into a fully valid asset. This makes the level of correspondence between the history recorded in the blockchain and the real history of the data in the distributed storage practically acceptable. As is known, ensuring full compliance of the real world history with the history recorded in a blockchain is outside the scope of the blockchain technology (the so-called the Oracle Problem, see e.g., https://medium.com/@DelphiSystems/the-oracle-problem-856ccbdbd14f). The compliance of ProvHL to the PMD standard W3C PROV [4], mentioned in Section 2, is provided by the conformity of the basic triples in the both data models, namely {Asset, Operation/Transaction, Participant} (for HLF&C/ProvHL) and {Entity, Activity, Agent} (for W3C PROV). A simplified scheme of the testbed with a preliminary version of the ProvHL system is presented in figure 1. The operational purpose of the HLF&C-network nodes (peers, OSNs, MSP modules) is specified in Section 3; the servers provide interaction with data management systems of local storages.

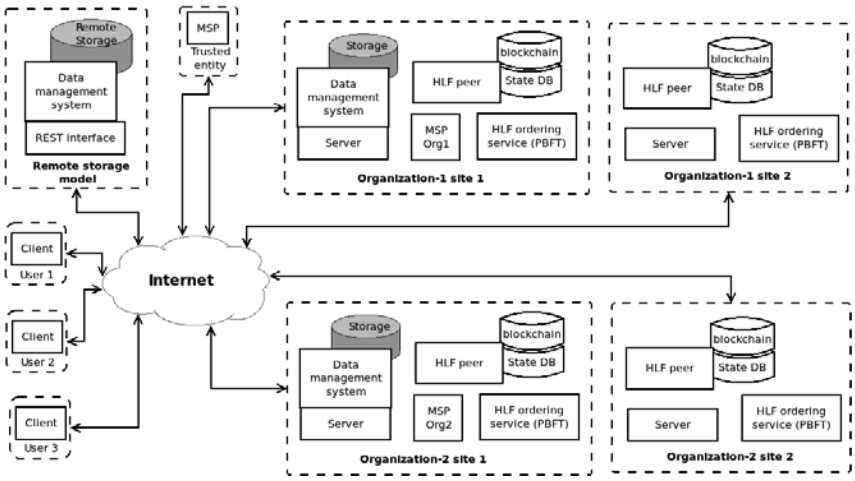

Figure 1. A simplified scheme of the ProvHL testbed.

6. Blockchain-Based Delegation of Rights in Distributed Storages

Delegation is the process of a user or a Web service handing over their authentication credentials to another executing Web service. We will present the mechanism of rights delegation between services on the example of the data copy operation from one local storage (Storage1) to another (Storage2). The definition of operations with files as the assets makes the mechanism of the delegation very natural and flexible. In the framework of the HLF&C-platform assets (as well as other business network entities) are defined with the help of an object-oriented modeling language [18] in the so called .cto-file. For the delegation mechanism it is important the operation definitions contain the obligatory attributes "requester" and "executor" as well as inherit "file owner" attributes from the file asset definition. Upon receiving a request from a User for a file copying, the DMS_Storage1 (the data management system of the Storage1 which contains the file to be copied) detects the type of the copy operation, namely decides if this is local copying (within the Storage1) or copying to another storage. In the latter case it initiates, on behalf of the User, the operation of uploading the required file to destination Storage2. For this aim it interacts with the chaincode which, among other actions, defines that: (a) while for the initial copy operation the value of the requester attribute is equal to the User and the executor is DMS_Storage1, for the induced upload operation the requester is DMS_Storage1 and the executor is DMS_Storage2; (b) the owner of the file copy on the Storage2 is the same as the owner of source file on the Storage1.

Thus, the second request is executed at the request of the User, but by the DMS_Storage1 (source storage), and the file ownership does not change. This means that all goals of a delegation are completed. It is worth stressing that in contrast to the scheme based on proxy certificates [19], in the blockchain-based approach the delegation is restricted solely to the specified operation.

7. Related Works

Although a number of projects have been implemented in recent years to create systems for supporting and managing metadata, including the provenance of data, but the vast majority of the implemented solutions are centralized (see the surveys [20, 21] and refs. therein), which is poorly consistent with the use of a distributed dynamically changing environment, and the possibility of using metadata by organizationally unrelated or weakly related research communities.

On the other hand, in recent years, distributed registries based on blockchain technology have become very popular in various applied areas due to a number of important advantages [9, 22]. Most recently, on the basis of blockchains, PMD management systems for storages

have also been developed. The system ProvChain [23] is designed to work with one cloud storage, not a distributed environment. The validation of the data provenance is done off-chain by a centralized provenance auditor. The system consists of five components: users, cloud service providers, blockchain network, provenance database, and the provenance auditor. The blockchain network consists of virtual nodes inside the cloud and keep the data provenance records in the blocks. The auditor retrieves the provenance data from the blockchain database and validates the blockchain receipt. The system is based on a bitcoin-like blockchain. To solve the issues of the initial distribution of ownership shares of resources during the operation of the Proof-of-Stake algorithm, as well as the validation of this share, a special service is used. This service starts with the beginning of the process of developing consensus and constantly works together with blockchain validators (the blockchain nodes being virtual machines in the cloud) to check their share of resources and select a leader in each round. This service also has the right to determine remuneration for the successful creation of new blocks and at the same time to punish (deprive a share of resources) if the validators act maliciously.

The SmartProvenance (DataProv) system [24] is implemented on top of the Ethereum blockchain platform and consists of two types of components. They are the blockchain components, which mainly consist of Ethereum smart contracts for access control, creating/storing PMDs and conducting the voting process, as well as modules not related to the blockchain including the client module that interacts with smart contracts and a script, working in the cloud, to check the validity of changes to data files. A smart contract Document Tracker is used to track changes in data and maintaining access control policies. Each PMD-related event, such as a modification of a document, must be approved by voting with a special smart Vote contract. The main purpose of the voting process is to prevent harmful changes that clearly violate the conditions of use of the data. At the end of the voting stage, if a change decision is made, the Vote contract submits a change in the Document Tracker contract to create a PMD record. Authorized voting users use a special verification script that is located in the cloud storage of the system. In the SmartProvenance system, the rejection of changes as a result of voting is punishable by a fine using smart contracts and cryptocurrency (Ether) of the Ethereum platform. On the contrary, voters who have discovered an attempt at unlawful changes receive a reward in the form of a certain amount in the cryptocurrency.

Actually both ProvChain and SmartProvenance/DataProv does not utilize full fledged blockchain technology and distributed consensus algorithms because though they use the blockchain registries to store the data provenance records, the blockchain nodes do not represent interests of the parties involved and the verification of the operations and the records is done outside the blockchain network by centralized provenance service. Therefore they do not really support business processes of data sharing between administratively unrelated parties. They also inherit the bitcoin/Ethereum platform properties which poorly matching the needs of a system for supporting business processes of data sharing in distributed storages. In particular, the both systems use permissionless blockchains with their inherent shortcomings with which we have to put up in the case of open blockchain networks but which are not inevitable in the case of distributed storages.

Other existing suggestions and developments in the field of PMD management, including those based on blockchains, are further from the ProvHL system proposed in this article, both in their goals and objectives, as well as in the ways of their implementations. The reader may find discussion of them in Section V of the survey [25].

Thus the traditional centralized techniques are mostly inefficient and have no specific protection for sensitive information [20, 21, 25]. The shortcomings and half-way solutions of the discussed blockchain-based systems show that the blockchain-based data provenance needed more research effort and can be further enhanced by proper utilization of the smart contracts. We believe that our ProvHL system meets the challenge.

8. Conclusion

In this paper, the new approach based on the integration of the blockchain technology, smart contracts and metadata driven data management was suggested. On its basis, the principles and algorithms of the system called ProvHL (Provenance HyperLedger) were developed. This system is a fault-tolerant, secure and reliable system for managing provenance metadata, as well as access rights to data in distributed storages. The problems of optimal choice of the blockchain type for such a system, as well as the choice of the blockchain platform were studied. Namely, it was proposed to use a permissioned type of blockchain and the Hyperledger blockchain platform, on the basis of which the ProvHL system is implemented.

Provenance metadata are written to the blockchain beforehand, and the data management system refers to the blockchain and performs the transactions recorded there (metadata driven data management). At present, a testbed has been created on the basis of SINP MSU, where a preliminary version of the ProvHL prototype is deployed to implement the developed principles and refine the algorithms of the system. Within the testbed, smart contracts are implemented to support basic operations with files (upload, download, copy to another storage, etc.). This infrastructure is free from the significant drawbacks inherent to existing solutions, in particular, from the vulnerabilities associated with the presence of a central services and the problems related to the using permissionless blockchains. In addition, the suggested blockchain-based delegation proves to be simple, natural and flexible. Creation of the ProvHL production level system will significantly improve the quality and reliability of the results obtained on the basis of data processing and analysis in a distributed computer environment.

References

1. Kryukov A. and Demichev A. Decentralized Data Storages: Technologies of Construction // Programming and Computer Software. 2018. Vol. 44, No. 5. P.303-315.
2. Berghöfer, T., et al. Towards a model for computing in european astroparticle physics. 2015. arXiv preprint, arXiv:1512.00988.
3. Androulaki E.,et al. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains // Proceedings of the Thirteenth EuroSys Conference, EuroSys'18, April 2326, 2018, Porto, Portugal, article No. 30. ACM, 2018.
4. PROV-DM: The PROV Data Model. W3C Recommendation. 2013. URL: https://www.w3.org/TR/prov-dm (accessed 03.06.2019).
5. Levy E. and Silberschatz A. Distributed File Systems: Concepts and Examples // ACM Computing Surveys. 1990. Vol. 22, No. 4. P.321-374.
6. Hamida E. B., et al. Blockchain for Enterprise: Overview, Opportunities and Challenges // The Thirteenth International Conference on Wireless and Mobile Communications, ICWMC2017,Nice, France, 2017.
7. Cachin C. and Vukolic M. Blockchain Consensus Protocols in the Wild. 2017. arXiv preprint, arXiv:1707.01873.
8. Valenta M., Sandner P. Comparison of Ethereum, Hyperledger Fabric and Corda. FSBC Working Paper. 2017. PP. 1-8.
9. Baliga A. Understanding Blockchain Consensus Models. Tech. rep., Persistent Systems Ltd. 2017.
10. Lamport, L. The Part-Time Parliament.// ACM Transactions on Computer Systems. 1998. Vol 16. No. 2. PP. 133-169.

11. Ongaro, D. and Ousterhout J.K. In search of an understandable consensus algorithm // Proceedings of the USENIX Annual Technical Conference, USENIX Association, 2014. PP. 305-319.

12. Howard H. ARC: analysis of Raft consensus. 2014. Technical Report No. UCAM-CL-TR-857. University of Cambridge.

13. Castro M. and Liskov B.: Practical Byzantine Fault Tolerance // Proceedings of the 3rd Symposium on Operating Systems Design and Implementation, 1999. PP. 173-186.

14. Sousa, J., Bessani, A. and Vukolic, M. A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform // Proceedings of the 48th annual IEEE/IFIP international conference on dependable systems and networks, 2018. PP. 51-58. IEEE.

15. Cachin C, Schubert S, Vukolić M. Non-determinism in byzantine fault-tolerant replication. 2016. arXiv preprint, arXiv:1603.07351

16. Liu S, et al. XFT: Practical Fault Tolerance beyond Crashes // Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI16), Savannah, GA, USA, 2016. PP. 485-500,

17. Baird, L. The Swirlds hashgraph consensus algorithm: Fair, fast, Byzantine fault tolerance. Swirlds Tech Report SWIRLDS-TR-2016-01. 2016. URL: https://www.swirlds.com/whitepapers (accessed 03.06.2019).

18. Hyperledger Composer Modeling Language, URL: https://hyperledger.github.io/composer/v0.19/reference/cto_language.html (accessed: 03.06.2019).

19. Tuecke S, et al. Internet X.509 Public Key Infrastructure Proxy Certificate Profile. 2004. Tech. Rep. RFC 3820.

20. Zafar F., et al. Trustworthy Data: A Survey, Taxonomy and Future Trends of Secure Provenance Schemes // Journal of Network and Computer Applications. 2017. Vol. 94. PP. 50-68.

21. da Cruz S. M. S., Campos M. L. M. and Mattoso M. Towards a Taxonomy of Provenance in Scientific Workflow Management Systems // Proceedings of the World Conference on Services-I, IEEE, 2009. PP. 259-266.

22. Azaria A., Ekblaw A., Vieira T., Lippman A. MedRec : Using Blockchain for Medical Data Access and Permission Management // Proceedings of the 2nd International Conference on Open and Big Data, OBD, Vienna, Austria, Aug. 2016. PP. 25-30.

23. Liang X. et al. Provchain: A Blockchain-based Data Provenance Architecture in Cloud Environment with Enhanced Privacy and Availability // Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, IEEE Press, 2017. PP. 468-477.

24. Ramachandran A. and Kantarcioglu M. SmartProvenance: A Distributed, Blockchain Based Data Provenance System // Proceedings of CODASPY'18: The 8th ACM Conference on Data and Application Security and Privacy. Tempe, AZ, USA, 2018. PP. 1-8.

25. Salman, T., Zolanvari, M., Erbad, A., Jain, R. and Samaka, M. Security services using blockchains: A state of the art survey // IEEE Communications Surveys & Tutorials. 2018. Vol. 21, No. 1. PP. 858-880.