Blockchain-Based Delegation of Rights in Distributed Computing Environment

Andrey Demichev[1(✉)], Alexander Kryukov[1], and Nikolai Prikhod'ko[2]

[1] Skobeltsyn Institute of Nuclear Physics,
Lomonosov Moscow State University, Moscow, Russia
{demichev,kryukov}@theory.sinp.msu.ru
[2] Yaroslav-the-Wise Novgorod State University, Velikiy Novgorod, Russia
niko2004x@mail.ru

Abstract. The paper suggests a new approach based on blockchain technology and smart contracts to delegation of rights within distributed computing systems, which is fault-tolerant, safe and secure. The implementation of the proposed approach is based on the permissioned blockchains and on the Hyperledger Fabric blockchain platform in conjunction with Hyperledger Composer.

Keywords: Distributed computing · Blockchain · Access rights · Delegation · Hyperledger

1 Introduction

Nowadays, distributed computing systems (DCS) are widely used for solving various problems in scientific, engineering and business areas. The advantage of DCS is the unification and simplification of an access to computing resources, e.g., clouds, supercomputers, databases, and, as consequence, to growth of efficiency of scientific, engineering and business activities. However, using heterogeneous and geographically widely dispersed DCS requires sophisticated and robust solutions for various aspects of the distributed computation in comparison with the case of local resources or more localized DCS. In particular, a reliable but still user-friendly security model for such DCS is of great importance. In this paper we discuss some aspects of the security infrastructure for DCS and suggest possible improvements. Providing the security of DCS implies solving the following basic problems: (1) security of communications: this problem is solved by encrypting the communication channels; (2) authentication: this means confirmation of the truth of the attribute of the data fragment declared by a certain entity as a true one; (3) authorization: this means the granting of access rights according to a policy; (4) delegation: this means delegation of rights from a user or a Web service to another Web service.

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

V. Malyshkin (Ed.): PaCT 2019, LNCS 11657, pp. 408–418, 2019.
https://doi.org/10.1007/978-3-030-25636-4_32

In this paper we consider the last aspect of the DCS security. We will use grid infrastructures and distributed storages as a reference DCS models for implementation of the security infrastructure. However the same problems are relevant and the suggested solutions are applicable for any DCS which comprises of a set of communicating Web services. The most striking example of grid infrastructure and globally distributed storage is the Worldwide LHC Computing Grid (WLCG) [1,2] which is used for processing and simulation of experimental data from the Large Hadron Collider (LHC) [3]. Other important examples of DCSs are the data storages and processing infrastructures in the area of astroparticle physics [4,5].

The security of most of DCSs, including WLCG, is based on the PKI [6] and X.509 certificates [7] together with proxy certificates [8]. The proxy certificate is a special short-time living certificate used for the purpose of providing restricted rights delegation within a PKI based authentication system. The short lifetime of the proxy certificate is due to security reasons. In DCSs the proxies are used for both user access to computing resources and for processing workflows. A workflow is a composite computational job that must be run sequentially by multiple services, with each service in the sequence receiving requests directly from the previous service. In this case, the delegation of rights from service to service occurs with the help of the proxy certificates. However, the proxies have short lifetimes, while one cannot predict how much time would take request processing especially in the case of the composite jobs. There are special services to support prolongation of proxy lifetime [9], and all this make the security infrastructure overcomplicated and difficult to interact with.

Recently, we proposed an approach [10,11] which allows us to avoid using the proxy certificates in security infrastructures entirely. Roughly speaking, in our scheme each issued request is a pair of a message and individual hash related to it. This single-shot hash has unlimited lifetime so that in our scheme the prolongation service is not needed. At the same time, the security level is not reduced because every hash can be used only once and only for a specific request. Thus hash compromise can only result in the fact that the request has to be processed again. However this approach also requires a central dedicated service, namely validation service, to process requests in DCSs. The point is that upon getting computational request each service checks request's hash against the validation service and continues only if the hash is correct and was not used before. Both the proxy prolongation service and the validation service in the approach suggested in [10,11] being centralized ones are potential points of failure and bottlenecks for the entire distributed systems.

In this work, we suggest a DCS design which allows abandoning the special dedicated centralized services in the DCS security infrastructure and the use instead of them a blockchain-based distributed registry and smart contracts. The very idea of using the blockchain technology for DCS security was expressed in our work [12]. However that paper does not contain any details of the design and is oriented to the Ethereum blockchain platform [13] which is not well suited for DCSs. In the present paper, we propose an approach to solving the problem

of delegation on the basis of blockchain technology and smart contracts within the Hyperledger platform [14,15] which is proved to be very suitable for DCS management, in particular for distributed storages [16]. While in the paper [16] we proposed a mechanism for managing provenance metadata and data access rights based on the blockchain technology, in the present work we solved another problem, namely, developing on the same basis a mechanism for delegation of rights in distributed systems. To our best knowledge, the blockchain-based mechanism for delegation of rights in distributed system suggested in this work are completely novel. Other existing blockchain-based suggestions and developments in the field of DCS management are far from the system proposed in this paper, both in their goals and objectives, and in the ways of their implementation. The reader may find discussion of them in the survey [17].

In the next section we shortly consider security infrastructure with the use of proxy certificates and the solution without proxy certificates but with a special central service. In Sect. 3 the blockchain-based delegation of rights in DCS is presented. The Sect. 4 is devoted to conclusions.

2 DCS Security Infrastructure

2.1 Security Infrastructure with the Use of Proxy Certificates

In distributed grid-like systems the security infrastructure is build around Public Key Infrastructure (PKI) that uses asymmetric cryptography. One of the main problem of the security infrastructure is the problem of delegation of rights [18,19]. Let us consider the delegation procedure in DCS for the following workflow (see Fig. 1): a Client asks the Service1 to perform a request; the Service1 sends a subrequest to Service2. It is expected that the Client somehow delegates its rights to Service1 to authenticate it to the Service2 since subrequest is performed on his behalf. Therefore there is a question how this delegation is carried out.

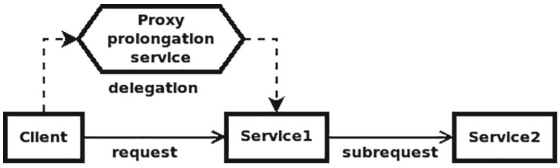

Fig. 1. Delegation of credentials.

The common solution used in grid is to use the proxy certificate with non-critical extension to store information about user rights. The proxy certificate is an extended X.509 public key certificate and has the following properties: it is signed with standard X.509 or another proxy certificate of a user who needs delegation of rights; contains both public and private keys; these are not the original users keys but generated from them; does not require any password (unlike

usual PKI certificates); cannot be revoked; is used by grid services, to act on behalf of the proxy issuer. Thus the proxy certificates are essentially less secure objects than standard certificates. To reduce the chance for proxy certificate to be stolen, the proxy must have very short lifetime. This leads to the problem of the renovation of the proxy. The possible solution of the problem is to use certain service that have to manage proxy certificates and renew them if necessary. One of such services is the MyProxy service [20].

The delegation scheme in this case looks as follows: (1) the user creates a proxy certificate; (2) it sends it to the service with a request to perform some action on behalf of the user; (3) from the point of view of any service, having a proxy certificate means that its bearer is authorized to do whatever it likes on behalf of the issuing the proxy. The last item leads to a vulnerability of the proxy certificate approach, namely, the service that received the proxy is given too much leeway on behalf of the entity issuing the proxy certificate. This is in addition to the above mentioned necessity to have the proxy prolongation service which is a potential point of failure, intrusion and bottle neck.

An example of a delegation is copying of a file from Service1 to Service2. For this aim a user transmits to Service1 his proxy certificate and requests it to copy a file to Service2 on his behalf so that the rights to the file will belong not to Service1 or Service2, but to the user. In Sect. 3.2 we will consider this use case for the delegation in the framework of the blockchain-based approach.

2.2 Intermediate Solution: Security Infrastructure Without Proxy Certificates and with Special Central Service

In the papers [10,11] a new security infrastructure model for distributed computing systems was suggested which does not require the proxy certificates. The proposed architecture of the DCS security infrastructure is shown in Fig. 2 on the left hand side.

Each request processed in DCS is accompanied by an accounting information. Accounting information is a triple of the following objects: $\{h, Entity_s, Entity_d\}$, where $h, Entity_s, Entity_d$ are the hash, source and destination entity of the request. This triple means that the entity $Entity_s$ sends a request with the hash h to the entity $Entity_d$ for execution. Complete format of accounting information include some additional objects such as affiliation to a virtual organization and user's roles in it.

Let us consider the processing of a request from the point of view of the credential delegation.

1. The Client generate a request r_1 and the hash $h_1 = H(r_1)$.
2. The Client registers the triple $\{h1, Client, Service1\}$ in the validation service (VS).
3. The Client sends the request r_1 to the Service1 for processing.
4. The Service1 generates the hash from the obtained request r_1 and asks the VS to approve it. If VS approves then Service1 continues.

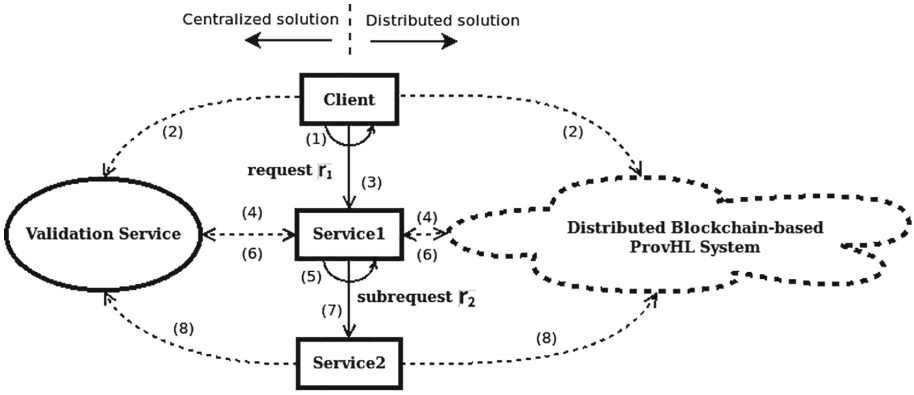

Fig. 2. The architectures of the security infrastructure with the central validation service and with the distributed registry (blockchain).

5. The Service1 generates the new subrequest r_2 that is generated from r_1 and the hash $h_2 = H(r_2)$.
6. The Service1 registers the triple $\{h2, Service1, Service2\}$ in the VS.
7. The Service1 sends the request to the Service2 for further processing.
8. The Service2 generates the hash from the obtained request r_2 and asks the VS to approve it. If VS approves then Service2 continues.

When Service1 registers $\{h2, Service1, Service2\}$, VS, knowing that this is a secondary request generated from the user's one, registers it as a user request. Thus, when accessing it by Service2, it will confirm that the action should be performed on behalf of the user, although received from Service1.

The hash of secondary requests should be calculated not only on the basis of the body of the new request, but also the hash of the primary request (a weak variant of the Merkle tree) from which it is generated. In processing the request, the validation service accumulates chains of accounting information for each request in the DCS. This information can be used for different purposes. In particular, it may be used for revocation of the request at any stage of processing.

One of the possible weak points of the proposed approach is the requirement to have on-line access to the validation service for all other services of the DCS. The simulation using our prototype shows that such an infrastructure is quite stable and works fine at least for the systems with 20 user requests per second. For the critical high-availability systems it is possible to deploy two parallel validation services with on-line database replication. At this case one of the services acts as a master service that processes requests and another is a slave (an inactive full copy of the master). If the master service crashes it would be easy to switch to the slave service immediately with almost no loss of information. An important benefit of the proposed security infrastructure is that during request processing the validation service collects all the information concerning each request in the DCS. This information can be used for monitoring purposes as well as for request revocation at any stage of processing.

3 Use of the Blockchain Technology for Providing Delegation of Rights in DCS

The approach shortly presented in Sect. 2.2 results in essential simplification both registration of new users in the system, and their operations in DCS, in comparison with the most popular infrastructure of public keys (PKI) together with use of the proxy certificates (Sect. 2.1). However the vulnerable point of both the solutions is need of a special fault-tolerant and resistant to malicious operations centralized service in the security infrastructure. In this section, we investigate the possibility to refrain from the special server in the security infrastructure of DCS and to use for this purpose a distributed registry based on the blockchain technology and smart contracts. Since in this case the security infrastructure registry is distributed across a number of nodes in the system, such an approach will lead to increased fault tolerance and level of security of DCS. The basic example of DCS which we use in present work is a distributed storage.

3.1 Distributed Storage with Provenance Metadata Driven Data Management

In the work [16] we proposed a new approach to the construction of data management systems in a distributed environment, based on the integration of the following basic principles and technologies:

- smart contracts [21];
- permissioned blockchains technology [22];
- Hyperledger blockchain platform [14,15] together with Hyperledger Composer [23]; hereafter we shall refer to these two components as HLF&C-platform;
- management of data access rights with the help of special HLF&C-platform tools;
- provenance metadata driven data management: the metadata is written to the blockchain beforehand, and data management systems (DMS) refer to the blockchain and performs the transactions recorded there;
- distributed consensus protocols [22].

Provenance metadata (PMD) contain key information that is necessary to determine the origin, authorship and quality of relevant data, their storage and usage consistency, and for interpretation and confirmation of relevant results of data processing. The need for PMD is especially important when data is jointly processed by several research groups that have their own, although interrelated interests, which is a very common practice in many scientific, engineering, and industrial fields lately. For the details we refer to the work [16] where principles, architecture and operation algorithms have been developed for the PMD management system, entitled ProvHL (Provenance HyperLedger), which is fault-tolerant, safe, reliable in terms of the safety and security of provenance metadata records from accidental or intentional distortion. Moreover, it allows users to perform operations with files and directories in the DCS. The distribution

of the main HLF&C modules by administrative domains of the modeled distributed storage environment within the current testbed for the ProvHL system is shown in Fig. 3. Here we shall concentrate on a new blockchain-based method for delegation of rights within distributed computing systems which is free from shortcomings inherent in other solutions.

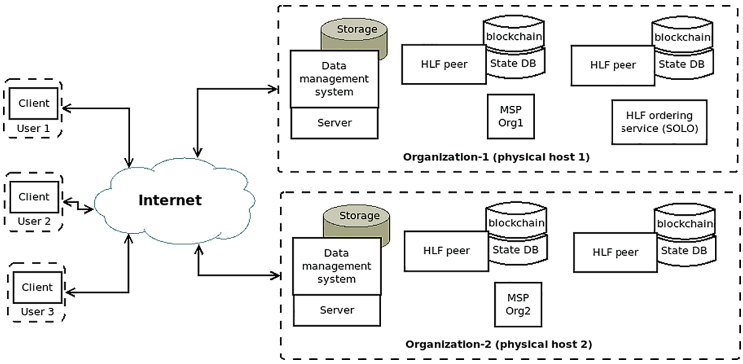

Fig. 3. A simplified scheme of the ProvHL testbed environment.

3.2 Blockchain-Based Delegation in Distributed Storages

The algorithm which we propose for recording transactions with provenance metadata and data management in the framework of ProvHL in a very simplified form reads as follows:

- the owner accesses the chaincode function, which, according to the acl-file ("acl" stands for access control language), allows the owner of the data to grant access rights to these data to another user or group of users;
- a user who is granted access rights by the owner accesses the chaincode with a request to make an operation (Client Request transaction) with data (for example, file download, upload, copy, etc.);
- the chaincode verifies that such a transaction complies with the rules defined in the acl-file and, if it does, sends a request to the HLF&C environment to complete the transaction;
- HLF&C performs transaction processing (transaction workflow: simulation and endorsements → ordering → validation → state updating);
- HLF&C sends a message (event) to the user about the successful transaction and its recording in the blockchain; the message also contains the transaction identification number;
- the user accesses the data management system (DMS) with a request to perform a data operation that contains the number of the corresponding transaction;
- the DMS checks for a record of this transaction in the blockchain;
- if there is a record of the valid transaction, the DMS performs the required operation and, in turn, initiates a transaction record confirming that a data operation was performed (Server Response transaction).

As it can be seen, for each data operation, at least two transaction records are made in the blockchain: one corresponds to the client request, and the second corresponds to the server response. Actually, an operation comprises of even more transactions.

Below we present more details on delegation of rights between services on the example of operation of coping data from one local storage (Storage1) to another (Storage2). Now the Service1 on Fig. 2 stands for the data management system of the Storage1 (DMS_Storage1) and Service2 stands for the data management system of the Storage2 (DMS_Storage2) and we use the right hand side of the figure (distributed solution). Now the content of the request r_1 is: "copy file F from Sorage1 to Storage2" and that for the request r_2 is: "upload file F to Storage2". In the framework of the ProvHL system, operations with files are defined as assets (alongside with other business network entities) [16] by using the object-oriented modeling language [23] in the so called cto-file. For the delegation mechanism it is important that it contains the obligatory attributes "requester" and "executor". Also it inherits "file owner" attributes from the file asset definition. Upon receiving a request from a User for a file copying the DMS_Storage1 (Storage1 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 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. In addition, the owner of the file copy on the Storage2 is the same as the owner of source file on the Storage1.

Note that in this case it is not necessary to rely on request hashes, as described in Sect. 2.2. Instead, one can use an arbitrary UUID for the request naming, since an immutability of record for a request sequence is guaranteed by the blockchain structure. The analog of the steps outlined in the Sect. 2.2 reads as follows.

1. The Client (User) generate the request r_1 and UUID for it.
2. The client initiates a transaction to create a copy operation, after which the entire transaction workflow is executed.
3. The Client sends the request r_1 to the DMS_Storage1 for processing. At this stage the 'requester' field of the operation attributes is equal to the Client, and the 'executor' is the DMS_Storage1.
4. The DMS_Storage1 checks that the related transaction is recorded in the blockchain and valid; in the case of positive result it continues carrying out the operation.
5. The DMS_Storage1 generates the new subrequest r_2 to DMS_Storage2 for uploading the file F to Storage2.
6. The DMS_Storage1 initiates recording the corresponding transaction into blockchain. At this stage the 'requester' field of the operation attributes is equal to the DMS_Storage1, and the 'executor' is the DMS_Storage2. It is

worth stressing that the right to initiate this request for the transaction is provided by the appropriate content of the smart contract (chaincode).
7. The DMS_Storage1 sends the request r_2 to the DMS_Storage2 for the file F uploading.
8. The DMS_Storage2 checks that the related transaction is recorded in the blockchain and valid. In the case of positive result it carries out the request.

Thus, the second request r_2 is executed at the initial request of the User, though it is issued 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 mentioning that in contrast to the scheme based on proxy certificates (Sect. 2.1), in the blockchain-based approach, as well as in the mechanism presented in the Sect. 2.2, the delegation is restricted solely to the specified operation. The chain of hashes used in Sect. 2.2 is replaced with a chain of transactions and blocks that make up the history of the copy operation from one storage to another. It is important to note that during the execution of the entire operation, the file F preserves the attribute "file owner" unchanged, that is, the rights to it in the process of the operations carried out by the chain of services (in this case, DMSs) do not change.

The approach proposed in this section allows us to avoid central services that can be bottlenecks, points of failure, and which are controlled by one of the sides of the business process. Instead, a distributed registry (blockchain) is used, which is controlled by all parties of the business process based on a consensus. The flexibility of the proposed mechanism is achieved due to the fact that in smart contracts one can fix any conditions for the delegation of rights. In this paper, we have considered a relatively simple, but in practice, most popular version of such conditions. The proposed mechanism directly extends to the case of arbitrary data processing services. Some technical complications are related to the fact that the result of such services can be an arbitrary number of output files. However, the general approach works in this case too.

The metric values of the developed system are under study and will be presented elsewhere. The preliminary measurements on the testbed depicted on Fig. 3 show that the overheads related to the operation processing by the ProvHL system is of the order of $4 \div 7$ s depending on setup variables such as maximal time of block forming, etc. This is fully consistent with the extensive results of the recent work [24] on the performance of the Hyperledger platform itself, with the measurements in this work were carried out on a testbed similar to ours. In particular, it was shown that for the input transaction rate up to 800 tx/s, the transaction latency is $\lesssim 1$ s, and the transaction throughput is $\sim$800 tx/s. If we take into account that each file operation consists of $3 \div 7$ transactions (depending on the type of the operation), we get matching results for the latency, while for the throughput we may expect $\sim$100 ops/s. These values, obtained on the testbed with very modest computer facilities, are quite acceptable for operations with files of sufficiently large volumes, the handing time of which (copying, downloading, uploading, etc.) is tens or more seconds. Such volumes of data files are typical for distributed storages intended for large scientific experiments.

4 Conclusion

In this work we have proposed a solution for a security infrastructure and delegation of rights for distributed computing systems based on the blockchain technology and smart contracts in the framework of the Hyperledger Fabric platform. This infrastructure is free from the significant drawbacks inherent to other existing approaches, namely, from the vulnerabilities (bottlenecks, points of failure) associated with the presence of a central services managing the security infrastructure. Due to its distributed nature, the blockchain-based delegation proves to be fully adequate to distributed computing systems. The use of smart contracts, in turn, provides flexibility because they allow one to define various conditions for the delegation of rights in DCSs.

At present, a testbed has been created on the basis of SINP MSU, where a preliminary version of the ProvHL system implementing the developed solution is deployed. Testing of the system has confirmed the correctness of the chosen approach, basic principles and algorithms of work and the preliminary performance measurements showed the suitability of the developed system for large distributed data storages.

The implementation of the suggested solution for delegation of rights in the framework of the ProvHL system of production level will significantly improve the security as well as quality and reliability of the results obtained on the basis of processing and analysis of data in a distributed computer environment.

References

1. Sciaba, A., et al.: Computing at the petabyte scale with the WLCG. Worldwide LHC computing grid. Technical report CERN-IT-Note-2010-006 (2010)
2. WLCG. http://wlcg.web.cern.ch. Accessed 21 May 2019
3. CERN. http://www.cern.ch. Accessed 21 May 2019
4. Berghöfer, T., et al.: Towards a model for computing in European astroparticle physics. arXiv preprint, arXiv:1512.00988 (2015)
5. Kryukov, A., Demichev, A.: Architecture of distributed data storage for astroparticle physics. Lobachevskii J. Math. **39**(9), 1199–1206 (2018)
6. Buchmann, J.A., Karatsiolis, E., Wiesmaier, A.: Introduction to Public Key Infrastructures. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40657-7
7. Cooper, D., et al.: RFC 5280 - internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. Technical report (2008)
8. Tuecke, S., et al.: Internet X.509 public key infrastructure proxy certificate profile. Technical report RFC 3820 (2004)
9. Kouril, D., Basney, J.: A credential renewal service for long-running jobs. In: IEEE/ACM Proceedings of the International Workshop on Grid Computing 13–14, pp. 2–13 (2005)
10. Dubenskaya, J., Demichev, A., Kryukov, A., Prikhod'ko, N.: Special aspects of the development of the security infrastructure for distributed computing systems. Procedia Comput. Sci. **66**, 525–532 (2015)

11. Dubenskaya, J., Demichev, A., Kryukov, A., Prikhod'ko, N.: New security infrastructure model for distributed computing systems. J. Phys. Conf. Ser. **681**, 012051 (2016)
12. Kryukov, A., Demichev, A.: Security infrastructure for distributed computing systems on the basis of blockchain technology. CEUR Workshop Proc. **1787**, 338–342 (2016)
13. A next-generation smart contract and decentralized application platform. White Paper. https://github.com/ethereum/wiki/wiki/White-Paper. Accessed 21 May 2019
14. Hyperledger. https://www.hyperledger.org. Accessed 21 May 2019
15. Androulaki E., et al.: Hyperledger fabric: a distributed operating system for permissioned blockchains. In: Proceedings of the Thirteenth EuroSys Conference, Article No. 30, Porto, Portugal. ACM (2018)
16. Demichev A., Kryukov A., Prikhod'ko N.: The approach to managing provenance metadata and data access rights in distributed storage using the hyperledger blockchain platform. In: Proceedings of Ivannikov ISPRAS Open Conference, Moscow, Russia, pp. 131–136, 22–23 November 2018. IEEE Xplore Digital Library (2019)
17. Salman, T., et al.: Security services using blockchains: a state of the art survey. IEEE Commun. Surv. Tutor. **21**(1), 858–880 (2018)
18. Welch V, et al.: 509 proxy certificates for dynamic delegation. In: 3rd Annual PKI R&D Workshop, vol. 14 (2004)
19. Smirnova O.: Grid computing: delegation and authorisation. http://www.hep.lu.se/courses/grid/2014/Grid-COMPUTE-l3.pdf. Accessed 21 May 2019
20. Novotny, J., Tuecke, S., Welch, V.: An online credential repository for the grid: myproxy. In: Proceedings of 10th IEEE International Symposium on High Performance Distributed Computing, pp. 104–111. IEEE (2001)
21. Szabo, N.: The idea of smart contracts (1997). http://szabo.best.vwh.net/smart_contracts_idea.html. Accessed 21 May 2019
22. Baliga, A.: Understanding blockchain consensus models. Technical report, Persistent Systems Ltd. (2017)
23. Hyperledger Composer. https://hyperledger.github.io/composer. Accessed 21 May 2019
24. Baliga, A., et al.: Performance characterization of hyperledger fabric. In: 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), pp. 65–74 (2018)