Public Auditing of Log Integrity for Cloud Storage Systems via Blockchain

Jia Wang[1], Fang Peng[1], Hui Tian[1(✉)], Wenqi Chen[1], and Jing Lu[2]

[1] College of Computer Science and Technology, National Huaqiao University, Xiamen 361021, China
{jwang, pengfang, htian, wqchen}@hqu.edu.cn
[2] Network Technology Center, National Huaqiao University, Xiamen 361021, China
jlu@hqu.edu.cn

Abstract. Cloud storage security has been widely focused by the industry and academia in recent years. Differing from the previous researches on cloud data integrity audit, we pay more attention to the security of log generated during the operation of cloud data. While cloud data is damaged and tampered by various security threats (e.g. faulty operations, hacker attacks etc.), it is one of the most common methods to track accidents through log analysis. Therefore, ensuring the integrity of the log files is a prerequisite for completing the incident tracking. To this end, this paper proposes a public model for verifying the integrity of cloud log based on a third party auditor. In order to prevent the log data from being tampered with, we aggregate the log block tags by using the classic Merkle hash tree structure and generate the root node which will be stored in the blockchain. In addition, the proposed scheme does not leak any log content during public audit. The theoretical analysis and experimental results show that the scheme can effectively implement the security audit of cloud logs, which is better than the past in terms of computational complexity overhead.

Keywords: Public auditing · Cloud forensics · Cloud security · Forensic investigation

1 Introduction

The various benefits of emerging cloud computing and its storage service have attracted a lot of enterprises and private users. Although it is popular with companies and private users, cloud storage can be targeted by criminals. As the users outsource their local data to the cloud, the security of data on the cloud becomes a very important issue [1]. Compared with the local environment, the dynamic nature of cloud computing makes traditional data security solutions become powerless, which brings greater challenges to the protection of cloud data [2–4]. Especially for users as the government, banking, securities, insurance and other industries, as well as large enterprise users, they attach great importance to data security. If sensitive data is stolen or leaked, it will cause huge losses to cloud users. Besides, there is another possibility that cloud service may be abused. For example, criminals can also store and hide incriminating and illegal

J. Li et al. (Eds.): SPNCE 2019, LNICST 284, pp. 378–387, 2019.
https://doi.org/10.1007/978-3-030-21373-2_29

materials in cloud, even distribute illegal information through shared cloud storage service [5].

Different to the previous cloud auditing research [6–8], we pay more attention to the security of log generated during the operation of cloud data. As any user behavior is recorded by cloud service provider as activity logs, the logs can be an important resource for extracting user behavior characteristics or illegal behavior. Collecting evidence of incidents or crimes involving cloud users and cloud services used by them is known as digital forensics (or cloud forensics) [9, 10]. Hence, logs are critical evidence in cloud forensics [11]. With analyzing logs, investigators can trace back after the event to obtain important clues [12]. However, acquisition of logs is completely dependent on the cloud service provider because logs are generated and stored by them. Considering that after an incident, the cloud service provider may modify the log data in order to evade responsibility. And the existence of some hardware and software failure factors will inevitably lead to the destruction of data. Therefore, the logs provided directly by the cloud service provider are not completely trusted. In our threat model, the cloud provider is honest when generates the log. However, after that the provider may hide the fact that logs have been tampered with, or even manipulate the logs for some business interests. An incompletely trusted log will throw subsequent investigations and forensics into a dilemma. To this end, ensuring the integrity of logs in cloud storage must be achieved before log analysis in digital forensics.

As trustworthy logs are very critical for digital forensics, researchers have come up with several solutions to ensure the security of log. However, most of current schemes focused on the secure logging method. Although some of them supported public verification, the schemes required downloading the whole logs for verification, which was not practical for cloud environment. In [13], public auditing for operation behaviors log in cloud storage was introduced, which enhanced the credibility of forensic results. However, the scheme was complex and caused large computational and communication overhead. On this basis, we made some improvements. In this paper, we present a novel public secure auditing scheme for logs in cloud based on blockchain. As blockchain can ensure the integrity and non-repudiation of data [14], the properties of blockchain make it suitable for protecting log security. However, the size of log in cloud is generally significant and it is not realistic to store log in the blockchain. Therefore, we just store only a small amount of important auxiliary information on the blockchain. In addition, considering that log information is highly sensitive and user's privacy issues are directly related to it, we use random masking technique proposed in [15], enabling the auditor to audit the logs without learning the log content. The proposed scheme ensures that the forensic investigators get secure and reliable logs.

The contributions of this paper are as follows:

1. We designed a secure and efficient auditing scheme for logs recording activities of users generated in cloud environment.
2. We introduce blockchain to prevent external attacker or CSPs from tempering with the logs after-the-fact.

3. We evaluate the performance of the proposed scheme by experiments and comparisons with the state-of-the-art schemes and prove our advantage in computing overhead.

The rest of this paper is organized as follow. Section 2 provides some background information and our public auditing model for logs in cloud storage. Section 3 presents our scheme in detail. In Sect. 4, we provide the performance evaluation, and finally we conclude in Sect. 5.

2 Problem Statement

2.1 Public Auditing Model for Logs in Cloud Storage via Blockchain

A representative public auditing model for cloud storage logs is illustrated in Fig. 1. The model includes three different entities as follows:

Users, an entity that needs to store large amounts of data in the cloud. The user is the owner of the data and has the right to share data (the data hosted by the user can be shared), which can be an individual or an organization.

Cloud Storage Provider (CSP), an entity that is responsible for supervising and maintaining the outsourced data provided by users. At the same time, in order to meet user needs, CSP also provides related services such as cloud computing and logging.

Third Party Auditor (TPA), an entity that is authorized by the user or its forensic institution to verify the integrity of log operations on the cloud and return the verification results.

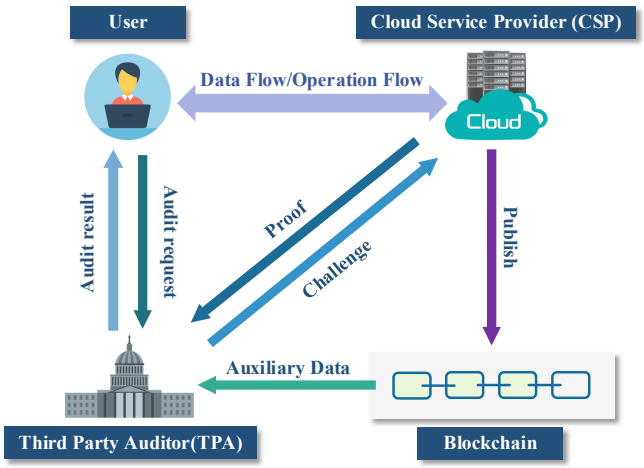

Fig. 1. Public auditing model for logs in cloud storage via blockchain

In general, users hosting large amounts of data on a cloud server can greatly reduce local storage and computational overhead. In addition, data stored on the cloud can be

easily shared with others. Generally, users and other data visitors who are granted legal rights can complete a series of operations on data through cloud services, such as adding, deleting, and modifying cloud data. And every operation of the data should be recorded in the log by the cloud service provider correctly, since there may be illegal operations on cloud data. Once the data is destroyed, we need to obtain evidence of illegal operations through log analysis. Determining the integrity of the log before log analysis is the key to solving the above problem. Therefore, we introduced a log integrity audit. In cloud data integrity auditing, a common practice is to introduce a trusted third party to improve the credibility of the audit process. In view of this, we propose a public audit model for logs in cloud storage based on third parties and use the blockchain as an aid.

2.2 Design Objectives

To efficiently support public verification of log integrity in cloud storage, our protocol is designed to achieve the following security and performance objectives:

- **Public auditing**: to enable any authorized TPA to verify the integrity of the cloud logs.
- **Auditing correctness**: to ensure that there exist no cheating cloud logs that can pass TPA's audit without indeed storing logs intact.
- **Privacy preserving**: to ensure that TPA cannot directly or indirectly obtain the actual log content during the auditing process.
- **Blockless verification**: to ensure that the actual log data does not need to be retrieved during the audit process.
- **Lightweight**: to enable TPA to carry out auditing with minimum communication and computation costs.

3 The Proposed Scheme

In this section, we first briefly introduce the preliminaries that will be used to construct the proposed scheme. Then, we present our public auditing for logs in cloud storage with all design details.

3.1 Preliminaries

Homomorphic Hash Function: A homomorphic hash function H is a collision-resistant hash function that for any vectors a, b and scalars α, β holds that $H(\alpha a + \beta b) = H(a)^{\alpha} H(b)^{\beta}$. Collision resistance implies that if one knows vectors $H(c) = H(a)^{\alpha} H(b)^{\beta}$ then it must be $c = \alpha a + \beta b$ [16].

Merkle Hash Tree (MHT): Merkle hash tree is a classic validation data structure [17] and is often used to verify the data integrity. As Fig. 2. Shows, it is a binary tree consisting of a root node, a set of intermediate nodes, and a set of leaf nodes. The bottom

leaf node contains the stored data or its hash value. Each intermediate node is the hash of the contents of its two child nodes as well as the root node [18], such as $h_d = h(h(N_3)||h(N_4))$. Suppose the verifier has the value h_r corresponding to the root node and he wants to verify the integrity of N_3 and N_6. Then, only the certifier needs to provide relevant auxiliary information $\Omega = \{N_3, N_6, h(N_4), h(N_5), h_c, h_f\}$. The verifier could use the auxiliary information to recursively obtain the root node $h_r^{'}$ by constructing the MHT and check whether the calculated $h_r^{'}$ is the same as the authentic one.

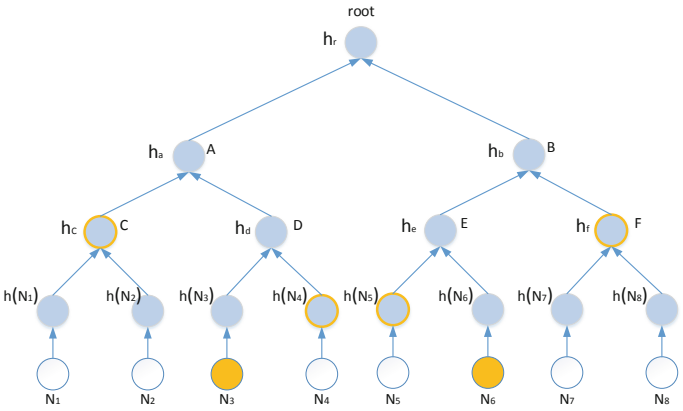

Fig. 2. Example of Merkle hash tree

Blockchain Technology: Blockchain [19] is a distributed general ledger technology derived from digital cryptocurrency bitcoin [20]. The development of this technology has aroused widespread concern in industry and academia. From a technical point of view, blockchain is a combination of innovations in various technologies, such as peer-to-peer network technology, asymmetric encryption technology, consensus mechanism, smart contracts, etc. Therefore, blockchain technology has many features and we have briefly summarized it here:

- Decentralization. A blockchain is a point-to-point network composed of many nodes. There is no centralized device and management organization. The maintenance of the network depends on all the nodes with maintenance functions in the network. Each node has equal status.
- Dependence. The blockchain operation rules and inter-node data are open and transparent, and nodes do not need to rely on trusted third parties to establish trust relationships in advance.
- Irreversibility. The latter block in the blockchain system stores the hash value of the previous block. Therefore, to tamper with the data of a certain block, the data content of the block and all subsequent blocks must be modified accordingly. So, tampering with data in a blockchain system is computationally infeasible.
- Traceability. The blockchain stores data by using a time-stamped chained block structure. So, the transactions on the chain are traceable.

3.2 Description of the Proposed Scheme

In our scheme, after each user operation is detected, a log entry e will be generated by CSP. The log entries generated within a certain period constitute a log block $B_i = \{e_{i1}, \ldots, e_{is}\}$. A log file F, which needs to be verified, consists of a fixed number of log blocks: $\{B_1, \ldots, B_n\}$. Let $H(\bullet)$ and $h(\bullet)$ denote a homomorphic hash function and a one-way hash function respectively, G be a multiplicative cyclic group of the prime order q, g represent the generator of G. The system public parameters are $\{G, q, g, H, h\}$.

Specifically, our scheme consists of 4 algorithms: {Setup, Challenge, Prove, verify}. The details of the proposed scheme are as follows:

1. **Setup Algorithm**: In this algorithm, the CSP generates a key pair. Then, CSP computes a tag for each log block B_i of the log file F, and constructs a Merkle tree with hashes of log block tags as leaf nodes. Finally, the CSP stores all the tags and publishes the tree root into blockchain to protect the integrity of the tags.
   a. The CSP generates tag σ_i for each log block B_i as:

$$\sigma_i = H\left(\sum_{j=1}^{s} e_{ij}\right) \tag{1}$$

   Let $\Sigma = \{\sigma_i\}$ be the set of tags, where $1 \leq i \leq n$.
   b. The CSP uses tags as leaf nodes to construct a Merkle tree MT and generates the tree root R. Finally, the CSP publishes the tree root R with blockchain to prevent the tags from being altered.
2. **Challenge Algorithm**: This algorithm is conducted by the TPA to generate an auditing challenge for checking the integrity of logs in the cloud. The TPA first randomly picks a set $L = \{l_1, \ldots, l_c\}$ with c elements, where $1 \leq l_i \leq n$. Then, the TPA chooses a random value $v_i \in \mathbb{Z}_p$, for each element $l \in L$. Finally, the TPA sends the auditing challenge $chal = \{i, v_i\}$ to the CSP, where $i \in L$.
3. **Prove Algorithm**: After receiving the auditing challenge, the CSP runs this algorithm to generate a proof of log integrity as follows:
   a. The CSP first chooses s random valuer $w_j \in \mathbb{Z}_p$, $1 \leq j \leq s$. Then, the CSP computes

$$W = H\left(\sum_{j=1}^{s} w_j\right) \tag{2}$$

$$E_j = \sum_{i \in L} v_i \cdot e_{ij} + w_j \tag{3}$$

   b. The CSP generates the sibling path from the leaf nodes $\{\sigma_i\}$ to the root node in the tree as an auxiliary information $\{\Omega_i\}$, where $i \in L$. Finally, the CSP responds the TPA with proof $P = \{W, E_j, \{\sigma_i, \Omega_i\}_{i \in L}\}_{j \in S}$.
4. **Verify Algorithm:** In this algorithm, the TPA first retrieves the root R from the blockchain. Then, the TPA computes root r using the received $\{\sigma_i, \Omega_i\}_{i \in L}$ and checks that if r and R are equal. If r is not the same value as R, the TPA outputs

False, which means that the tags of the log blocks stored in cloud has been tampered with. Otherwise, the TPA checks where the following equation holds:

$$W \cdot \prod_{i \in L} \sigma_i^{v_i} = H\left(\sum_{j=1}^{s} E_j\right) \tag{4}$$

If the equation holds, output *TRUE*; otherwise, False.

Here, we give the demonstration about the correctness of the verification Eq. (5) as follows:

$$\begin{aligned} \text{Right} &= H\left(\sum_{j=1}^{s}\left(\sum_{i \in L} v_i \cdot e_{ij} + w_j\right)\right) \\ &= H\left(\sum_{j=1}^{s}\sum_{i \in L} v_i \cdot e_{ij} + \sum_{j=1}^{s} w_j\right) \\ &= H\left(\sum_{i \in L} v_i\left(\sum_{j=1}^{s} e_{ij}\right)\right) \cdot H\left(\sum_{j=1}^{s} w_j\right) \\ &= \prod_{i \in L} H\left(\sum_{j=1}^{s} e_{ij}\right)^{v_i} \cdot W \\ &= \prod_{i \in L} \sigma_i^{v_i} \cdot W = \text{Left} \end{aligned} \tag{5}$$

4 Performance Evaluation

4.1 Theoretical Analysis

We compare our scheme with ref. [18] in terms of computational cost. For convenience, we define the following notations to denote the operations. *EXP* and *MUL* to denote the complexity of one exponentiation operation and one multiplication operation on Group *G* respectively. With the use of homomorphic hash function, our computation overhead in tag generation for each log file is *nEXP* operations, where *n* is the number of elements in block, compared with the overhead in [18] which is 2*nEXP* and *n* hash operation. Since both schemes use Merkle hash tree, we do not consider the time of constructing the MHT in this section. In our scheme, the CSP runs the *Prove* algorithm with one *EXP* and *c MUL* operations, where *c* is the number of challenged blocks. After that, TPA runs *Verify* algorithm with (*c* + 1) *EXP* and one *MUL* operations. However, in [18] needs *cEXP* and c*MUL* operations while the TPA needs two Pairing, 2*cEXP* and c*MUL* operations. In conclusion, our scheme can reduce computation overhead in both CSP and TPA side.

4.2 Experimental Results

In this subsection, we show the performance of our scheme by the experiments implemented with python based on Pairing Base Cryptography (PBC) library version 0.5.14 and employs an MNT d159 curve based on 160-bit group order to achieve 80-bit security parameter. We conduct all the experiments on a Linux system (ubuntu 16.04.2 LTS x64 with 4.8.0 kernel version) with an Intel Xeon E3-1225 v5 CPU at 3.31 GHz, 8 GB RAM and a 7200 RPMSATA 2 TB. All the results are the averages of 20 trials. In experiments, we assume that a log file is 20 MB composed by 5000 blocks.

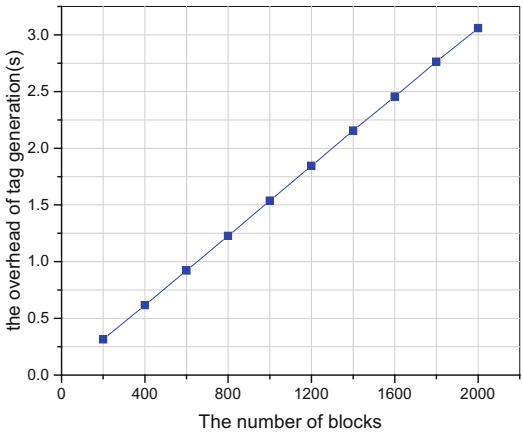

Fig. 3. The computation overhead of tag generation under different number of blocks

We conduct the experiments for tag generation and evaluate its performance in Fig. 3, where the number of blocks is increased from 200 to 2000 with intervals of 200. As we can see, the time cost of tag generation linearly increases with the number of blocks, which ranges from 315.56458 ms to 3059.623603 ms.

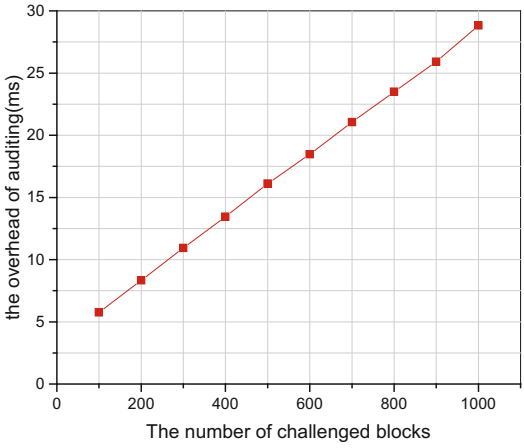

Fig. 4. The computation overhead of TPA in challenge phase under different number of challenged blocks

The computation cost for the algorithm challenge, proof and verify is shown in Figs. 4 and 5. The results of these experiments under different number of challenged blocks varied from 100 to 1000 show that the time in different auditing phases increases proportionally to the number of challenged blocks. From the above experimental results, it is clearly that most of the computation overhead is taken by the CSP and TPA only costs a little time. Therefore, our proposed scheme achieves high efficiency on TPA side.

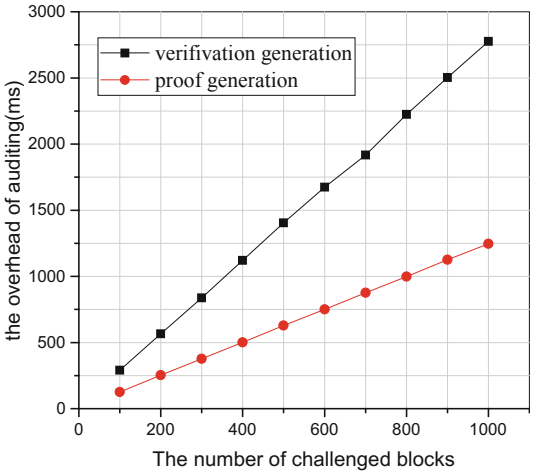

Fig. 5. Computation overhead in different auditing phase under different number of challenged blocks

5 Conclusion

In this paper, we present a public auditing scheme for logs in cloud storage with privacy preserving. Specifically, we utilize homomorphic hash function to generate tag of log block. As the tag is generated by the CSP and the CSP is not completely trusted, we aggregate the log tags by Merkle hash tree and publish the tree root with blockchain, which not only reduces the computation cost of CSP to generate tags, but also prevents the log tags from being tampered with. Besides, our proposed scheme can support public auditing without leaking the log content with the use of random masking technique. The theoretical analysis and experiment results show that the proposed scheme has a good performance.

Acknowledgement. This work was supported in part by National Natural Science Foundation of China under Grant Nos. U1405254 and U1536115, Natural Science Foundation of Fujian Province of China under Grant No. 2018J01093, Program for New Century Excellent Talents in Fujian Province University under Grant No. MJK2016-23, Program for Outstanding Youth Scientific and Technological Talents in Fujian Province University under Grant No. MJK2015-54, and Research Project for Young Teachers in Fujian Province (Program for High-Education Informationization) under Grant No. JAT170055.

References

1. Tang, J., Cui, Y., Li, Q.: Ensuring security and privacy preservation for cloud data services. ACM Comput. Surv. **49**(1), 1–39 (2016)
2. Wang, C., Wang, Q.: Security challenges for the public cloud. IEEE Internet Comput. **16**(1), 69–73 (2012)
3. Xiao, Z., Xiao, Y.: Security and privacy in cloud computing. IEEE Commun. Surv. Tutorials **15**(2), 843–859 (2013)
4. Puthal, D., Sahoo, B.P.S., Mishra, S.: Cloud computing features, issues, and challenges: a big picture. In: International Conference on Computational Intelligence and Networks. IEEE, Bhubaneshwar (2015)
5. Coileáin, D.Ó., O'mahony, D.: Accounting and accountability in content distribution architectures: a survey. ACM Comput. Surv. **47**(4), (2016). https://doi.org/10.1145/2723701
6. Tian, H., Chen, Y., Chang, C.: Dynamic-hash-table based public auditing for secure cloud storage. IEEE Trans. Serv. Comput. **10**(5), 701–714 (2017)
7. Wang, C., Wang, Q., Ren, K.: Toward secure and dependable storage services in cloud computing. IEEE Trans. Serv. Comput. **5**(2), 220–232 (2012)
8. Zhu, Y., Ahn, G., Hu, H.: Dynamic audit services for outsourced storages in clouds. IEEE Trans. Serv. Comput. **6**(2), 227–238 (2013)
9. Zawoad, S., Dutta, A.K., Hasan, R.: Towards building forensics enabled cloud through secure logging-as-a-service. IEEE Trans. Dependable Secure Comput. **13**(2), 148–162 (2016)
10. Martini, B., Choo, K.-K.R.: Cloud forensic technical challenges and solutions: a snapshot. IEEE Cloud Comput. **1**(4), 20–25 (2014)
11. Martini, B., Choo, K.-K.R.: An integrated conceptual digital forensic framework for cloud computing. Digit. Invest. **9**(2), 71–80 (2012)
12. Dykstra, J., Sherman, A.T.: Acquiring forensic evidence from infrastructure-as-a-service cloud computing: exploring and evaluating tools, trust, and techniques. Digit. Invest. **9**, S90–S98 (2012)
13. Tian, H., Chen, Z., Chang, C.: Enabling public auditability for operation behaviors in cloud storage. Soft. Comput. **21**(8), 2175–2187 (2017)
14. Dorri, A., Steger, M., Kanhere, S.S.: BlockChain: a distributed solution to automotive security and privacy. IEEE Commun. Mag. **55**(12), 119–125 (2017)
15. Wang, C., Wang, Q., Ren, K., Lou, W.: Privacy-preserving public auditing for data storage security in cloud computing. IEEE Trans. Comput. **62**(2), 362–375 (2013)
16. Xu, W., Feng, D., Liu, J.: Remote data integrity checking protocols from homomorphic hash functions. In: 14th IEEE International Conference on Communication Technology. IEEE, Chengdu (2012)
17. Ralph, C.: Merkle: protocols for public key cryptosystems. In: 1980 IEEE Symposium on Security and Privacy, pp. 122–122. IEEE, Oakland (1980)
18. Wang, Q., Wang, C., Ren, K.: Enabling public auditability and data dynamics for storage security in cloud computing. IEEE Trans. Parallel Distrib. Syst. **22**(5), 847–859 (2011)
19. Christidis, K.: Blockchains and smart contracts for the Internet of Things. IEEE Access **4**, 2292–2303 (2016)
20. Swan, M.: Blockchain: Blueprint for a New Economy. O'Reilly Media, Sebastopol (2015)