Comparison of Blockchain-Based Solutions to Mitigate Data Tampering Security Risk

Mubashar Iqbal(✉) and Raimundas Matulevičius

Institute of Computer Science, University of Tartu, Tartu, Estonia
{mubashar.iqbal,raimundas.matulevicius}@ut.ee

Abstract. Blockchain-based applications are arising because they ensure integrity, anti-tampering, and traceability. The data tampering risk is one of the main security concerns of data-centric applications. By the nature of the blockchain technology, it is befitting a revolutionary solution to mitigate the tampering risk. But there exists no proper guidance to explain how blockchain-based application could mitigate this risk. In this paper, we consider tampering risk management and discuss how blockchain-based applications could mitigate it. The study includes a comparison of different solutions.

Keywords: Blockchain · Security risks · Data tampering security risk · Security risk management · Security modelling

1 Introduction

Blockchain is a decentralised distributed and immutable ledger technology [1]. The use of blockchain technology ensures integrity, anti-tampering, and traceability [2]. The blockchain performs a consensus mechanism and data validation before saving on the immutable ledger. The blockchain-based application detects and discards all the unauthorised data changes during the consensus and data validation if the majority of the network is honest (i.e., not controlled by an adversary). This process establishes a tamper-proof environment [3].

Blockchain technology is emerging in different application domains to overcome various security challenges. Data tampering is the main security concern, which developers attempt to mitigate by blockchain-based solutions [4]. Data tampering involves the malicious modification of data by an unauthorised user [5]. Data exists in two states; either in transit or stored. In both cases, data could be intercepted and tampered [6]. Damage to the critical data could cause disruption to revenue-generating business operations. In the worst case scenario, it could put people life at risk, e.g., the tampering in the healthcare data [7].

Data becomes one of the most valuable assets in an organization. In order to help an organization to build secure software, various programs (e.g., OWASP

C. Di Ciccio et al. (Eds.): BPM 2019 Blockchain and CEE Forum, LNBIP 361, pp. 13–28, 2019.
https://doi.org/10.1007/978-3-030-30429-4_2

[8]) and threat modelling (e.g., STRIDE [9]) are working to communicate and reduce the tampering risk. Recently, the blockchain-based solutions are appearing to mitigate the data tampering risks [10,11]. In this paper, we follow the ISSRM domain model [12,13] and perform the data tampering risk management. The main objective is to compare the architectures for the blockchain-based solutions in order to explain how tampering risk could be mitigated. Hence, we consider (*i*) the assets to secure from the tampering risk, (*ii*) vulnerabilities, which cause the tampering risk, (*iii*) security requirements for risk treatment, and (*iv*) the potential countermeasures to mitigate the tampering risk. The main contributions of our work are: (1) data tampering risk analysis to identify what resources should be secured, (2) traditional technique-based countermeasure architecture to mitigate tampering risk, (3) Ethereum-based countermeasure architecture, (4) Hyperledger fabric-based countermeasure architecture to mitigate tampering risk, and (5) the comparison of countermeasure for the tampering risk.

The rest of the paper is structured as follows: Sect. 2 bestows a background and literature review. Section 3 presents the context and assets identification. Section 4 presents the mitigation of tampering security risk. Section 5 yields a comparison of tampering risk countermeasures. Section 6 provides the discussion and Sect. 7 concludes the paper.

2 Background

Blockchain is a peer-to-peer (P2P) network-based distributed ledger technology. It forms a chain by a sequence of blocks where each block is attached to the previous block by a cryptographic hash. Blockchain is classified as a permissionless or permissioned [14]. Permissionless blockchain allows anyone to join or leave the network and transactions are publicly visible. In permissioned blockchain, only predefined verified nodes can join the network and transactions visibility is restricted [14,15].

Ethereum platform is an example of permissionless blockchain. It uses the Ether cryptocurrency for the administration fee and proof of work (POW) consensus mechanism. Hyperledger fabric (HLF) is an example of permissioned blockchain and it follows the practical Byzantine fault tolerance (PBFT) based consensus mechanism. HLF uses permissioned settings to allow different participants to access a different set of data.

A system is secure whenever there is no possible way to attack it and it is less likely to be possible even with the blockchain technology. Blockchain helps one to overcome various security risks [4] and is acknowledged to be less vulnerable because of the decentralised consensus paradigm to validate the transactional information. The software security modelling can help to identify/visualize the security issues, and to uncover the hidden security needs. In this paper, we present the management of data tampering risk to explicate how the blockchain-based solutions are supporting the mitigation of this risk.

2.1 ISSRM Domain Model

In this paper, we follow an information systems security risk management (ISSRM) domain model [12,13]. ISSRM comprises three main concepts groups: (*i*) asset-related concepts, (*ii*) risk-related concepts, and (*iii*) risk treatment-related concepts. The asset could be classified as an IS system asset or business asset. The business asset has value and system asset (or IS asset) supports it. Security criteria (confidentiality, integrity and availability) distinguish the security needs. In risk-related concepts, the risk is a combination of risk event and impact. The risk event constitutes the threat and one or more vulnerabilities. The threat targets the IS asset and it is triggered by the threat agent, who uses an attack method and exploits the vulnerability. Impact harms the asset and negates the security criteria. The risk treatment presents a decision to treat the security risk and to define the security requirements. Security requirements are implemented as the controls (security countermeasures) to improve the security of the system.

2.2 Literature Review

In [4], we report on a literature review where security risks to blockchain-based applications are presented. The study explains what security risks of centralised application are mitigated and what security risks appear after introducing the blockchain technology. It also aggregates a list of possible countermeasures. The study categorises the findings by permissionless (i.e., Bitcoin, Ethereum & Customised permissionless), permissioned (i.e., Hyperledger fabric & Customised permissioned) and generic blockchain platforms. The results show that data tampering risk is one of the main security risks. In this study, we consider only the data tampering risk. Currently, Ethereum and HLF platforms provide the complete blockchain solution to build decentralised applications (dApps). Other blockchain platforms are also suitable to build dApps (e.g. EOS & R3 Corda etc.), but these are not yet widely adopted. Hence, we include only those literature studies where the Ethereum and HLF applications are considered to mitigate the data tampering risk.

Ethereum Solutions. In [16], authors illustrate how to protect the user preferences and privacy policies in the IoT systems. The authors of [17] present the blockchain solution in healthcare domain to protect the patient and medical data. In [18], secure mutual authentication scheme is discussed to protect the authentication credentials and response messages from the tampering risk. In the resource monitoring domain [19], the authors incorporate the blockchain-based authorisation system to secure the resource consumption data. Hjalmarsson *et al.* [20] utilize the blockchain to maintain the integrity of voting data and voting results by mitigating the tampering risk. Pop *et al.* [21] employ the blockchain solution as a security layer to protect the bidding and big-offer data.

Hyperledger Fabric Solutions. The authors [10,19] present a blockchain solution to protect the patient and medical data from being tampered. Yu *et al.* [22]

incorporate the blockchain solution as a security layer to protect the voting data from tampering risk. The study [11] builds the blockchain-based IoT video surveillance system to protect the videos recordings and settings from being tampered. In order to mitigate the drug counterfeit [23], the authors implement the blockchain-based solution.

In this paper, we will base our discussion on these studies (see Sect. 2.2). We will show the tampering risk context, its components and potential mitigation countermeasures.

3 Context and Assets Identification

In this section, we define the context and assets, which relate to the data tampering risk. Next, we analyse how tampering could harm the protected assets.

Table 1 shows assets secured from the tampering risk. It also presents the relationship between business assets, security criteria and system assets. For

Table 1. Assets and their security criteria

Paper	Business asset	System asset
HLF-based applications assets and security criteria		
[24]	Patient data (C, I), Healthcare data (I)	*Storage* (Healthcare data), *Service* (Store data), *Service* (Request data)
[22]	Voting data (I)	*Storage* (Voters data), *Storage* (Voting data), *Service* (Store data)
[10]	Patient data (C), Medical records (I)	*Storage* (Patient data), *Storage* (Medical records), *Service* (Store data)
[11]	Video recordings (I), Monitoring (A), CCTV settings (I)	*Storage* (Video recordings), *Storage* (CCTV settings), *Service* (Store data)
[23]	Drug certificate (I)	*Storage* (Drugs data), *Storage* (Supply chain data), *Service* (Store data)
Ethereum-based applications assets and security criteria		
[16]	User preferences (I), Privacy policies (I)	*Storage* (User preferences data), *Service* (Store data)
[17]	Patient data (I), Medical data (I)	*Storage* (Patient data), *Storage* (Medical data), *Service* (Store data)
[18]	Authentication (A), Response message (I)	*Storage* (Response message), *Service* (Manage access rights), *Service* (Store data)
[19]	Resource consumption data (I)	*Storage* (Resource consumption), *Service* (Store data)
[20]	Voting data (I), Voters data (C, I), Voting result (I)	*Storage* (Voting data), *Storage* (Voters data), *Storage* (Voting result), *Service* (Store data)
[21]	Bidding data (I), Bid-offer data (I)	*Storage* (Bidding data), *Storage* (Bid-offer data), *Service* (Store data)

example, the business assets (i.e., patient and healthcare data) are supported by the system assets (i.e., storage of healthcare data, service of store and request data). Security criteria (C - Confidentiality, I - Integrity, A - Availability) are constraints of the business assets.

The architecture, presented in Fig. 1, is an abstraction of the system assets defined from the literature study in Table 1. It characterises the system components at four layers. The *User Layer* exposes the users who interact with the application. The *Interface Layer* presents the various interfaces of the application. The user interacts with the services, which are present in a *Service Layer*. The *Data Storage Layer* shows the database.

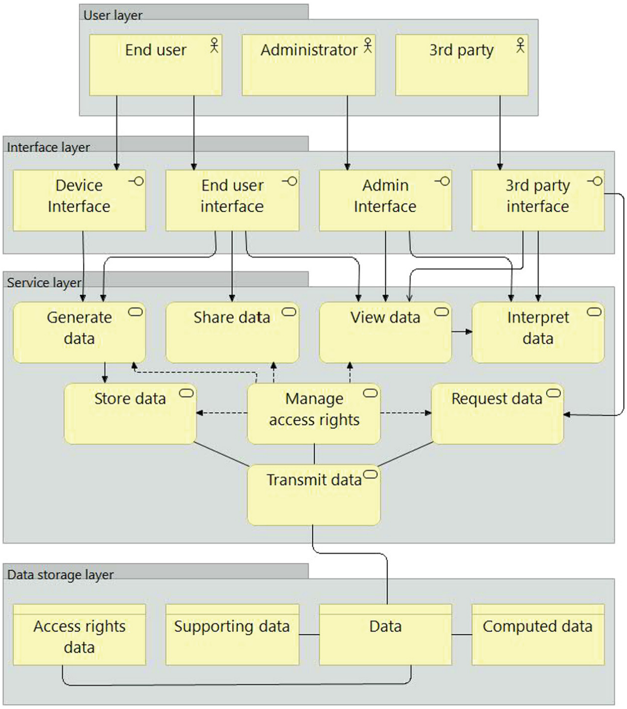

Fig. 1. Architecture of traditional system assets

In Fig. 2, an abstraction of the data tampering risk is presented. The details are collected from the literature studies. Figure 2 demonstrates the security risks, vulnerabilities and the main components of a traditional application. It helps to visualize the vulnerable system assets. The *Threat Agent (Attacker)* commands the *Data Tampering Threat* and leads to the *Data Tampering Risk*. *Risk* is a combination of *Threat* and *Vulnerabilities* that provokes a negative *Impact* and negates the *Security Criteria*. The vulnerabilities are connected to the system assets and depict their weaknesses. It allows an attacker to harm vulnerable system assets. The following vulnerabilities are obtained:

V#1: Lack of information validation
V#2: Lack of auditability
V#3: Lack of crypto functionality
V#4: Poorly implemented access control
V#5: Weak transmission channel

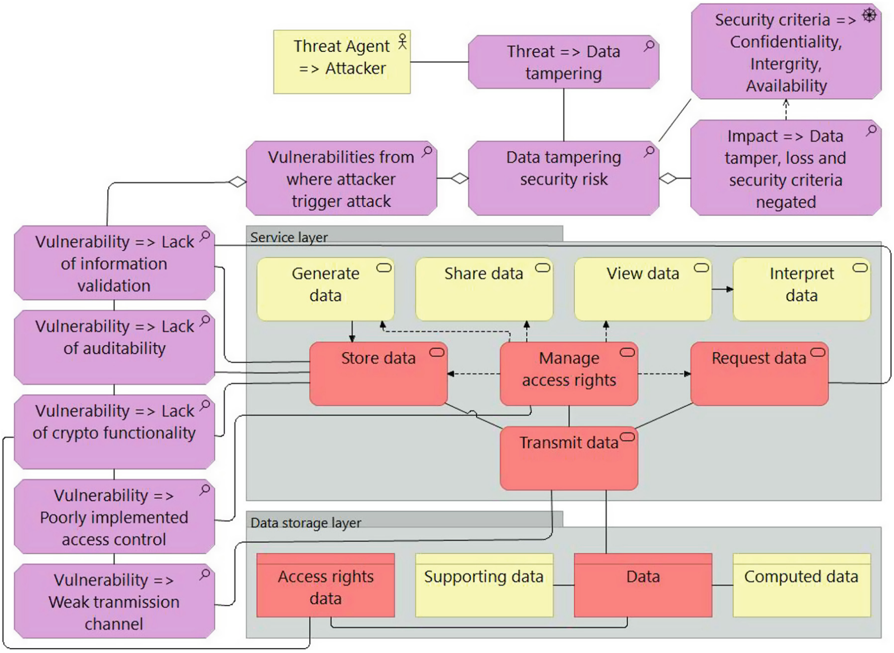
Threat Agent => Attacker
Threat => Data tampering
Security criteria => Confidentiality, Intergrity, Availability
Vulnerabilities from where attacker trigger attack
Data tampering security risk
Impact => Data tamper, loss and security criteria negated
Vulnerability => Lack of information validation
Vulnerability => Lack of auditability
Vulnerability => Lack of crypto functionality
Vulnerability => Poorly implemented access control
Vulnerability => Weak transmission channel
Service layer
Generate data
Share data
View data
Interpret data
Store data
Manage access rights
Request data
Transmit data
Data storage layer
Access rights data
Supporting data
Data
Computed data

Fig. 2. Architecture of data tampering security risk

For example, *Store data* service is vulnerable because there is a lack of information validation (V#1), lack of auditability (V#2) and lack of crypto functionality (V#3). *Manage access rights* service is vulnerable because of poorly implemented access control (V#4). Similarly, the *Request data* service is vulnerable due to a lack of information validation (V#1) and *Data storage* – due to a lack of crypto functionality (V#3) and weak transmission channel (V#5).

Based on the mentioned literature sources (see Sect. 2.2), the following security requirements (SR) are set to mitigate the tampering risk:

SR#1: The system should perform data validation
SR#2: The system should provide the data auditability
SR#3: The system should incorporate the crypto functionality
SR#4: The system should provide access control
SR#5: The system should provide a secure transmission channel

In the next section, we will discuss how these requirements are implemented to mitigate tampering risk using traditional countermeasures, and using the blockchain-based applications.

4 Mitigation of Tampering Security Risk

In order to address the mitigation of tampering risk, we present the three countermeasure architectures: (*i*) the traditional techniques-based countermeasure architecture, (*ii*) the permissionless blockchain-based countermeasure architecture following the *Ethereum* platform, and (*iii*) the permissioned blockchain-based countermeasure architecture following the *Hyperledger fabric* platform.

4.1 Traditional Countermeasure Architecture

Figure 3 shows how the identified vulnerabilities are mitigated by the traditional countermeasure techniques. The data tampering threat is represent in the STRIDE threat model [9], which has the corresponding set of security countermeasures (SC) to reduce tampering threat:

SC#1: Validate and filter input data
SC#2: Create secure audit trails
SC#3: Incorporate the crypto functionality and use digital signatures
SC#4: Use strong authorisation and access control
SC#5: Secure communication with protocols

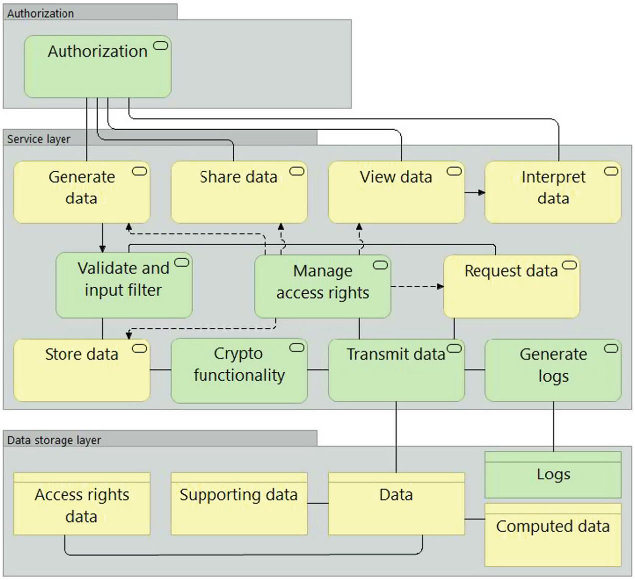

Fig. 3. Traditional techniques-based countermeasure architecture

Based on these countermeasures the architecture (see Fig. 3) is build to exhibit the countermeasures components which are applied to reduce tampering risk. For example, the security countermeasure (SC#1) employs on the Store data and Request data to mitigate the vulnerability related to lack of information validation (V#1). The countermeasure (SC#2) regarding the secure audit trails helps to mitigate the lack of auditability vulnerability(V#2) of Store data. The countermeasure (SC#3) of crypto functionality and the use of digital signatures approach mitigates the lack of crypto functionality vulnerability (V#3) of Database and Store data. The countermeasure (SC#4) mitigates the poorly implemented access control vulnerability (V#4). The countermeasure (SC#5) helps to mitigate the weak transmission channel vulnerability (V#5).

4.2 Ethereum-Based Countermeasure Architecture

Ethereum platform provides immutable decentralised distributed ledger, which ensures tamper-proof recording of transactions [17]. Along with the blockchain solution, Ethereum-based decentralised applications introduce several other techniques, which we consider as countermeasures. These Ethereum-based countermeasures (EC) are collected from the literature studies (see Table 1), which are utilized to secure an application. These countermeasures also help to clarify the security needs of Ethereum application:

EC#1: Transaction data validation [18]
EC#2: Store an encrypted data on the immutable ledger [16,17,20,23]
EC#3: Blockchain-based access control [16,21]
EC#4: Split the data and store in random locations [17]

Ethereum-based application ledger is distributed among peers. Because of this, it is impossible to remove the data from all the peers. Also, tampering is impossible because of the blockchain nature, which validates the information before recording it on the ledger. Furthermore, the attacker cannot execute the malicious code because it is impossible to control all the peers simultaneously unless the attacker controls 51% of the mining power. This is impossible to achieve for him because of the current mining difficulty and Ethereum ledger maturity.

The architecture (see Fig. 4) incorporates the identified countermeasures along with the Ethereum blockchain solution to mitigate tampering risk. The countermeasure (EC#1) mitigates the vulnerability related to lack of information validation (V#1). The lack of auditability vulnerability (V#2) is mitigated by an immutable ledger of the blockchain. The countermeasure (EC#2) approach mitigates the lack of crypto functionality vulnerability (V#3). The countermeasure (EC#3) mitigates the poorly implemented access control vulnerability (V#4). The weak transmission channel vulnerability (V#5) is not mitigated directly but it is controlled by P2P network, access control, data validation and consensus. For example, access control only allows those users who have specific access rights. If an application does not implement access control then invalid transmitted data is discarded on data validation and consensus stages. Data

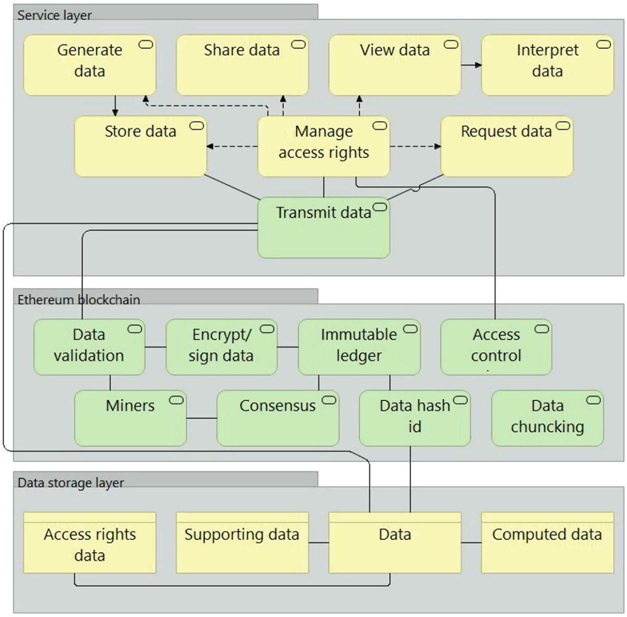

Fig. 4. Ethereum-based countermeasure architecture

chunking (EC#4) is used to deals with limitation of large file storage on the ledger but it also provides tampering resistance. The data file chunks are stored on several random locations along with unique hash id and indexes. If an adversary tampers the chunk then it invalidates the hash and that particular data chunk becomes invalid.

4.3 Hyperledger Fabric-Based Countermeasure Architecture

As compare to Ethereum-based solutions, HLF solves performance, scalability, and privacy issues by permissioned mode of operation and fine-grained access control. Likewise, HLF-based decentralised applications introduce several other techniques to mitigate tampering risk, that we consider as countermeasures. These HLF-based countermeasures (HC) are collected from the literature studies (see Table 1). These countermeasures also help to clarify the security needs of HLF application:

HC#1: Transaction data validation [23, 24]
HC#2: Traceability of ledger transactions [10]
HC#3: Store an encrypted data on the immutable ledger [11, 22, 23]
HC#4: Blockchain-based access control [24]

In Fig. 5 the suggested countermeasures are illustrated. HLF introduces a PBFT based consensus mechanism. It performs the data validation and writes the records on the immutable distributed ledger.

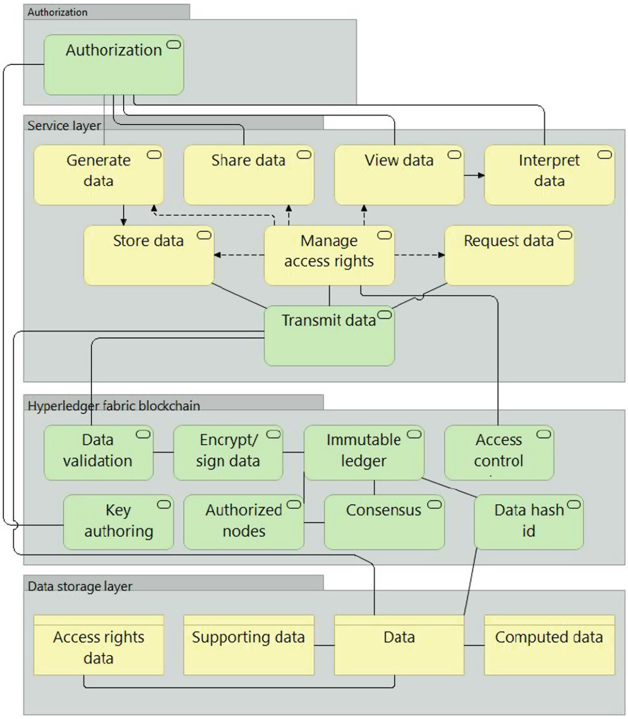

Fig. 5. HLF-based countermeasure architecture

The architecture (Fig. 5) incorporates the identified countermeasures along with the HLF blockchain solution to mitigate tampering risk. The countermeasure (HC#1) mitigates the vulnerability related to a lack of information validation (V#1). The lack of auditability vulnerability(V#2) is mitigated by an immutable ledger of the blockchain and traceability of ledger transactions (HC#2). The countermeasure (HC#3) mitigates a lack of crypto functionality vulnerability (V#3). The countermeasure (HC#4) mitigates the poorly implemented access control vulnerability (V#4). Similarly to Ethereum case, the weak transmission channel vulnerability (V#5) is not mitigated directly but it is controlled by P2P network, access control, data validation and consensus.

5 Comparison

In this section, we compare the above-mentioned approaches by their potential vulnerabilities and respective countermeasure techniques.

Table 2 lists the vulnerabilities and their respective countermeasures used in different solutions. For example, to mitigate V#1, traditional application implements centralised data validation and input filtering before recording in the database. The Ethereum-based application performs distributed data validation

Table 2. Comparison of different solutions which mitigate data tampering risk

	Traditional	Ethereum	HLF
V#1	Centralised validation and filter input data	Distributed data validation by unverified nodes [18]	Distributed data validation by verified nodes [23,24]
V#2	Audit trails	Tamper-proof immutable distributed ledger [16,17,20,23]	Tamper-proof immutable distributed ledger, and traceability of ledger transactions [10]
V#3	Crypto functionality and digital signatures	Blockchain-based crypto functionality, hashing and digital signatures [16,17,20,23]	Blockchain-based crypto functionality, hashing and digital signatures [11,22,23]
V#4	Authorisation and access control	Blockchain-based access control [16,21]	Predefined verified nodes, Blockchain-based access control [24]
V#5	Secure communication with protocols	Split the data and store in random locations [17], and encrypted data communication [17,23,25]	Encrypted data communication [24–26]

by unverified nodes called miners. Miners validate the data and only record in tamper-proof immutable ledger if valid. Similarly, HLF performs distributed data validation by verified nodes. As mentioned above HLF is a permissioned blockchain and the participant nodes are verified. These mitigation techniques have benefits and limitations against one another. For instance, the traditional application performs faster data validation but it lacks the full control [27]. As the data validation and filtering rule are centralised and written by developers, they could be error prone [27,28]. Ethereum performs data validation through validator nodes [29] by checking the data against the defined validation rules, including historical data in the ledger. Ethereum provides a transparent platform to define data validation rules which agreed upon by other nodes. Then, all the nodes follow those rules to validate the incoming data. Also, blockchain is an append-only structure and user can only add data but can not modify or delete them [28]. Hence, this process reduces human error. But POW is an energy-waste consensus mechanism. It takes time to validate and also pays an administration fee to the miners for performing this activity. These limitations are overcome by HLF which does not require POW or administration fee. It uses the PBFT consensus for data validation. By the nature of HLF, it leverages the benefits of permissionless blockchain (e.g., Ethereum) as well as it provides faster, inexpensive, efficient and privacy-oriented data validation [24].

In order to mitigate V#2, traditional application separately implements functionality for keeping audit trails (aka logs). Audit trails provide transparency and proof for records integrity and accuracy. It also protects sensitive data from an intentional misuse or harm from involving parties in the business process. The audit trails in the traditional application are weak, vulnerable and subject to attacks [30]. The control remains to a designated authority in a traditional centralised approach. It does not provide transparent traceability and trust-able proof of audit trails integrity. In contrast, blockchain-based decentralised application manage records in an immutable ledger, which provides tamper-proof transparent audit trails with backward traceability [27]. In Ethereum, whenever a new transaction occurs it appends on the ledger and replicates among nodes over P2P network. Similarly, HLF provides the immutable ledger and rich traceability of ledger transactions [10].

The third vulnerability (V#3) is mitigated by incorporating the crypto functionality and digital signatures. The traditional application integrates crypto functionality to save data securely. Again, it lacks control over data. Since the centralised authority is responsible for an administration of the database and if the security is compromised then the attacker can steal, modify or remove the data. It does not matter if the data is stored is an encrypted format or not. These attacks are common in centralised traditional application [31]. Ethereum and HLF allow to save encrypted data on the ledger, so it becomes possible for a client node to encrypt the data before submitting a transaction. The records are difficult to modify or delete because of the consensus mechanism and ledger redundancy among nodes over P2P network and also because of an append-only structure of the blockchain. As Ethereum is a permissionless blockchain and anyone can read the data from the ledger, it is possible for an attacker to trigger deanonymization (linking) attack. In contrast, HLF overcomes this limitation by verified nodes and permissioned settings of the ledger.

The fourth vulnerability (V#4) is mitigated by implementing authorisation and access control. It is a security control [32] to check who can access the system and data. In a traditional application, authorisation and access control settings could be tampered because of centralised storage and weak auditing. Ethereum-based access control settings are hard to tamper because those are validated by the nodes. Also, the settings are distributed among nodes which makes impossible for an attacker to change on all the nodes. In Ethereum, it requires an extra effort/work to implement access control. In HLF, only verified nodes are allowed to participate in the network. It also provides fine-grained access control to share specific access rights among various nodes.

The last vulnerability (V#5) is related to the weak transmission channel. In the traditional application, it mitigated by providing secure communication with protocols that ensures the integrity of transit data. The weak implementation of communication protocols could be broken [33]. In this case, an attacker can intercept data transmission and modify the data. Ethereum overcomes this issue by splitting the data and storing them in random locations with their respective indexes. It also provides encrypted data communication. In the Ethereum-based P2P system, nodes can send and receive data directly from each other, also behave both as a server and as a client [34]. In Ethereum, the valid transaction is usually signed before submitting but the associated data is not encrypted by default [35]. In this case, the client node encrypts the transaction data and then submits it to the network [25]. The acting server node knows that the transacting data is correct and valid because of validation and consensus process [27]. Let's suppose, if an attacker tampers the data then it would not be validated during the validation and consensus process. Similarly, HLF provides encrypted data communication and validates the transit data.

6 Discussion

Even though implementing the STRIDE countermeasures to protect from tampering risk, the traditional approaches lack full control over data security. For example, the attacker could get access to the database, could tamper or delete it. The attacker could trigger a ransomware attack and encrypt the database. The attacker could send the malicious code and tamper the record. He leaves no traces because of the weak audit trails. The application has a weak authorisation and access control. Crypto functionality is not properly implemented. These are only a few limitations which counter by the traditional application.

Here it comes the blockchain solutions, which record each transaction in a tamper-proof immutable ledger. Blockchain supports an append-only ledger and saves every transaction with a unique cryptography hash. The consensus mechanism and validator nodes validates incoming data. The immutable ledger provides rich transparency, audibility and traceability. It ensures that the records on the ledger are accurate and unaltered. Ethereum is a permissionless blockchain platform for building a decentralised application. In some cases, Ethereum-based application is also not feasible; for example, a bank/financial or healthcare application where data visibility and privacy is critical. Ethereum platform is based on the permissionless blockchain so the ledger is publicly accessible. It is also expensive because of the administration fee and energy-waste POW consensus mechanism. In this case, permissioned blockchain-based solution is a feasible choice, for example, the application of the permissioned HLF platform.

Our current study has a few limitations. For example, the current approach has a limited number of literature sources which address mitigation of data tampering risk as comparing it to the existing ones. In this work, we performed a subjective literature based comparison. In general, the blockchain technology looks promising in the perspective of organisation security, but it is still in its infancy. There are not many blockchain applications in production to assess the security and their countermeasures on a larger scope. By overcoming these limitations could bring richer insights and enhancement in this paper results.

7 Concluding Remarks

In this work, we present a comparison of different solutions to mitigate the data tampering risk. More specifically we considered: (*i*) traditional techniques-based solutions, (*ii*) permissionless Ethereum-based solutions and (*iii*) permissioned

HLF-based solutions. Results of the study could be considered when evaluating the software design in the perspective of tampering risk to produce secure software.

Apart from the tampering risk, blockchain-based applications could help mitigating other security risks [4], like DoS/DDoS attack, MitM attack, side-channel attack and etc. However, the blockchain-based applications are not a *silver bullet*: for instance, a number of security risks (e.g., sybil attack, double spending attack, 51% attack and other) are among the frequently observed ones in the literature [4]. We plan to compare different solutions to mitigate them in future research.

As a part of the future work, we plan to develop a blockchain-based comprehensive security risk reference model in order to systematically evaluate the overall security of the blockchain-based application. The model would not be dependent on the specific blockchain type or blockchain platform. It would be generic enough to cover the other security risks and blockchain platforms.

Acknowledgement. This research has been supported by the Estonian Research Council (grant IUT20-55).

References

1. Sato, T., Himura, Y.: Smart-contract based system operations for permissioned blockchain. In: 2018 9th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2018 - Proceedings 2018-Janua, pp. 1–6 (2018)
2. Chen, L., Lee, W.K., Chang, C.C., Choo, K.K.R., Zhang, N.: Blockchain based searchable encryption for electronic health record sharing. Future Gener. Comput. Syst. **95**, 420–429 (2019)
3. Tosh, D.K., Shetty, S., Liang, X., Kamhoua, C.A., Kwiat, K.A., Njilla, L.: Security implications of blockchain cloud with analysis of block withholding attack. In: Proceedings of 17th IEEE/ACM International Symposium on Cluster. Cloud and Grid Computing, CCGRID 2017, pp. 458–467 (2017)
4. Iqbal, M., Matulevičius, R.: Blockchain-based application security risks: a systematic literature review. In: Proper, H., Stirna, J. (eds.) CAiSE 2019. LNBIP, vol. 349, pp. 176–188. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20948-3_16
5. Microsoft: Transaction Integrator Threat Mitigation (2017)
6. Study.com: What is Data Tampering? - Definition & Prevention
7. Fimin, M.: Five early signs of data tampering (2017)
8. Khan, M.A., Salah, K.: IoT security: review, blockchain solutions, and open challenges. Future Gener. Comput. Syst. **82**, 395–411 (2018)
9. Ruffy, F., Hommel, W., Eye, F.V.: A STRIDE-based security architecture for software-defined networking. In: ICN 2016, The Fifteenth International Conference on Networks, no. c, pp. 95–101 (2016)
10. Chen, J., Ma, X., Du, M., Wang, Z.: A blockchain application for medical information sharing. In: TEMS-ISIE 2018–1st Annual International Symposium on Innovation and Entrepreneurship of the IEEE Technology and Engineering Management Society, pp. 1–7 (2018)

11. Gallo, P., Quoc Nguyen, U.: BlockSee: blockchain for IoT video surveillance in smart cities Suporn Pongnumkul NECTEC Thailand. In: 2018 IEEE International Conference on Environment and Electrical Engineering and 2018 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe), pp. 1–6 (2018)
12. Dubois, É., Mayer, N., Heymans, P., Matulevičius, R.: Intent. Perspect. Inf. Syst. Eng. **2010**, 1–384 (2014)
13. Matulevičius, R.: Fundamentals of Secure System Modelling, 1st edn. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-319-61717-6
14. Pradeepkumar, D.S., Singi, K., Kaulgud, V., Podder, S.: Evaluating complexity and digitizability of regulations and contracts for a blockchain application design. In: 2018 ACM/IEEE 1st International Workshop on Emerging Trends in Software Engineering for Blockchain, no. 1, pp. 25–29 (2018)
15. Ali, S., Wang, G., White, B., Cottrell, R.L.: A blockchain-based decentralized data storage and access framework for PingER. In: Proceedings - 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications and 12th IEEE International Conference on Big Data Science and Engineering, Trustcom/BigDataSE 2018, pp. 1303–1308 (2018)
16. Cha, S.C., Chen, J.F., Su, C., Yeh, K.H.: A blockchain connected gateway for BLE-based devices in the Internet of Things. IEEE Access **6**, 24639–24649 (2018)
17. Li, H., Zhu, L., Shen, M., Gao, F., Tao, X., Liu, S.: Blockchain-based data preservation system for medical data. J. Med. Syst. **42**, 1–13 (2018)
18. Lin, C., He, D., Huang, X., Choo, K.K.R., Vasilakos, A.V.: BSeIn: a blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0. J. Netw. Comput. Appl. **116**(February), 42–52 (2018)
19. Alcarria, R., Bordel, B., Robles, T., Martín, D., Manso-Callejo, M.Á.: A blockchain-based authorization system for trustworthy resource monitoring and trading in smart communities. Sensors **18**(10), 3561 (2018)
20. Hjalmarsson, F.P., Hreioarsson, G.K., Hamdaqa, M., Hjalmtysson, G.: Blockchain-based e-voting system. 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), pp. 983–986 (2018)
21. Pop, C., et al.: Decentralizing the stock exchange using blockchain an ethereum-based implementation of the Bucharest stock exchange, pp. 459–466 (2018)
22. Yu, B., Liu, J.K., Sakzad, A., Steinfeld, R., Rimba, P., Au, M.H.: Platform-Independent Secure Blockchain-Based Voting System, vol. 2433. Springer, Heidelberg (2018)
23. Sylim, P., Liu, F., Marcelo, A., Fontelo, P.: Blockchain technology for detecting falsified and substandard drugs in distribution: pharmaceutical supply chain intervention. J. Med. Internet Res. **20**(9), e10163 (2018)
24. Bhuiyan, Z.A., Wang, T., Wang, G.: Blockchain and big data to transform the healthcare, pp. 2–8 (2018)
25. Li, J., Wu, J., Chen, L.: Block-secure: blockchain based scheme for secure P2P cloud storage. Inf. Sci. **465**, 219–231 (2018)
26. García-Magariño, I., Lacuesta, R., Rajarajan, M., Lloret, J.: Security in networks of unmanned aerial vehicles for surveillance with an agent-based approach inspired by the principles of blockchain. Ad Hoc Netw. **86**, 72–82 (2019)
27. Dai, H., et al.: TrialChain: a blockchain-based platform to validate data integrity in large. Biomed. Res. Stud. 1–7 (2018)
28. Ray, S.: Blockchains versus Traditional Databases (2017)
29. Dexter, S.: How Are Blockchain Transactions Validated? Consensus VS Validation (2018)

30. Owasp: Top 10–2017 A10-Insufficient Logging & Monitoring (2017)
31. Domain, C.P.: From Yahoo to Uber, major hacks of data
32. Mellado, D., Blanco, C., Sánchez, L.E., Fernández-Medina, E.: A systematic review of security requirements engineering. Comput. Stand. Interfaces **32**(4), 153–165 (2010)
33. Rao, U.H., Nayak, U.: Understanding Networks and Network Security, pp. 187–204. Apress, Berkeley (2014)
34. Dagan, G.: The Actual Networking behind the Ethereum Network: How It Works (2018)
35. Pozo, A.: Ethereum: Signing and Validating (2017)