Ensuring Genuineness for Selectively Disclosed Confidential Data using Distributed Ledgers: Applications to Rail Wayside Monitoring

Matthias Lohr* *matthiaslohr@uni-koblenz.de*
Jonathan Hund* *hund@uni-koblenz.de*
Jan Jürjens*† *http://jan.jurjens.de*
Steffen Staab*‡ *staab@uni-koblenz.de*

*CCRDMT University of Koblenz-Landau, Germany †Fraunhofer ISST Dortmund, Germany ‡University of Southampton, United Kingdom

***Abstract*—In railway incidents, data from sensors installed on railway tracks can help finding the cause of the incident and identifying the responsible parties. Since the data collected may contain business-relevant information, it is usually treated as confidential by the companies collecting it. However, this data can only be considered as evidence if it can be proven that the data is genuine and unaltered, even if it is only accessible for involved companies in the first place.**

In this paper, we present an approach to ensure the genuineness of confidential railway measurement data using distributed ledgers and describe an approach for selectively sharing parts of the data without compromising confidentiality or the verifiability of genuineness. We also discuss how our approach can be generalized beyond the railway domain to show how distributed ledger-based approaches can be used to ensure the genuineness of confidential and selectively shared data.

***Index Terms*—railway, distributed ledger, blockchain, data integrity, merkle tree**

I. Introduction

In order to prevent accidents and breakdowns during the operation of railways, sensors are installed on the tracks to collect information of the condition of passing trains. This information helps rolling stock operators (planning and conducting the rides of trains, wagons, ...) to carry out timely maintenance operations or infrastructure managers (operating and maintaining fixed assets like sensors, tracks, signals, ...) to stop defective trains in order to prevent major damage or accidents. Typically, sensors of this kind are part of the infrastructure and are purchased by and installed on behalf of the infrastructure managing company, may be operated by another company (the supplier of the sensors), and collects data linked to the rolling stock of yet others companies (the different operators of the various rolling stock passing the sensors). This creates a complex situation where data is created and shared across company borders, but only selectively so (e. g. the competing rolling stock operators should not see their competitors confidential data).

In the case of an incident like a defect or an accident, the question of responsibility will come up. In many cases, incidents in the railway sector result in high costs [1][1]. Therefore, involved parties have a strong incentive to assign the responsibility to other parties or at least to be able to claim their own innocence. One way of doing this could be forging the measurement data. Without further protection, this is relatively simple, as the data is distributed to the relevant infrastructure managers and rolling stock operators and stored on their systems (see figure 1 and background information in section III), including the one which is interested in forging the data. Measurement data determined by the sensor is first transferred to a central database of the Infrastructure Manager. Subsequently, it is forwarded to the Rolling Stock Operator responsible for operating the measured train. On request, he forwards data to Rolling Stock Owners. If one of these parties unilaterally forges the data, it is easy to identify the differences in the data of several parties, but there is currently no way of distinguishing between original and the forged data.

In order to be able to use the measurement data as evidence, it is necessary to be able to ensure its genuineness, independently from the storage location and the parties involved. Furthermore, the genuineness has still to be verifiable even if only parts of the data are shared.

In this paper, we discuss the possibility of how distributed ledger-based approaches can be used to ensure the genuineness of confidential and selectively shared data. Although we have mainly used examples from the field of railway infrastructure, our concept is not limited to this application. It can be applied to any scenario of confidential data, where at the time the data set is created, it is not yet known to the creator of the data (e. g. the sensor) how the data will be partitioned for sharing.

We focus on distributed ledger technology because it allows us to exclude the dependency on central services or institutions like a single Trusted Third Party.

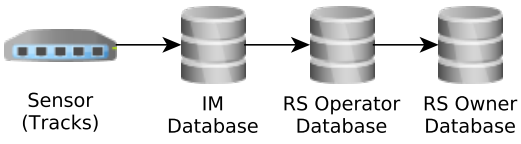

Fig. 1: Measurement data flow. IM: Infrastructure Manager, RS: Rolling Stock

[1] `http://ccrdmt.uni-koblenz.de/` – Global Railway Accident Database and Evaluation

In section II we will provide an overview of the related work and we give some background information about our example domain in section III. In section IV we describe our research objectives and will then present our approach in section V. The evaluation and discussion of our approach can be found in section VI. Further information on how our approach can be generalized is outlined in section VII. We then conclude with section VIII summarizing our results and describe possible future work.

II. Related Work

The research area of distributed ledger technology is based on the idea of a blockchain [2], a concept, presented by Nakamoto, for the decentralized management of a ledger. In general, the blockchain concept is considered as a solution to the Byzantine Generals Problem [3], which in simple terms means that malicious behavior by individual parties has no negative effect as long as a sufficient majority can agree on a common approach. This concept is used to distribute control over usually centrally controlled services (bank, notary, ...) to a community in which the misconduct of individuals has no impact on the general public. The *crypto currency* Bitcoin is a first example of how a centrally controlled concept such as that of a currency (e. g. controlled by the respective national central banks) can be secured against manipulation attempts by individual parties through the use of blockchains or comparable concepts.

The original data structure of a blockchain consists of a single chain of blocks which are linked by a cryptographic hash pointing to the respective preceding block, adapting the concept of Linked Timestamping which was described already in 1990 by Haber et al. [4]. This idea of a distributed Linked Timestamping is used in blockchain networks such as Bitcoin [2] and Ethereum [5] in a distributed context. Approaches such as Byteball [6], IOTA (Tangle) [7] or Hashgraph [8] replace the internal linear chain with a more complex directed acyclic graph.

Based on this property, there are already frameworks [9], concepts [10] and web services[2] for ensuring data integrity using distributed ledger technology. These concepts can also be applied on databases as shown in [11].

In the context of distributed ledgers, Merkle Trees [12] are often used to increase the efficiency of verification and hash-protected information. Instead of a hash which is computed by sequential processing of the data to be hashed, the hash (also denoted with *root hash*) is computed by dividing the data into usually $2^n, n \in \mathbb{N}$ parts, computing a hash for each part and then building a binary, balanced tree by bottom-up creating nodes by hashing two sub-level hashes.

In Bitcoin [2] Merkle Trees are used to ensure integrity and validity of transactions. For that, a Merkle Tree is created for a list of transactions. To verify a specific transaction within this list, the transaction itself and certain nodes from the Merkle Tree are sufficient for proving the integrity and validity of this specific transaction. In OriginStamp [10], Merkle Trees are used to combine multiple entries before they got added to the blockchain to reduce transaction fees.

In this paper we focus on the use of Merkle Trees under the aspect of confidentiality. As far as we know, there is no dedicated work that specifically focuses on this aspect.

The idea of a so-called *support chain*, which we will explain later, was already used in [13].

III. Background Information: Measurement Values of Rail Wayside Monitoring

For the detection of defective trains, there are various types of stationary sensors, each of which detects certain characteristics of the passing train These sensors are installed directly into the tracks and record individual values per wheel, axle or wagon. All sensors of a measuring point are connected to one or more measurement computers, which are located in small houses or switch cabinets next to the tracks and carry out the initial processing of the sensor signals.

Train rides are typically organized and carried out by rolling stock operators. Particularly for a freight train, it is typical that the wagons of a train belong to different owners who have commissioned the rolling stock operator to carry out the ride. This means that when a train is measured, objects from different owners are recorded in one dataset. As part of the infrastructure, the sensors first transfer this data to the IT systems of the infrastructure manager (first to the measurement computers nearby the sensor and from there to the central IT infrastructure of the infrastructure manager). Typically, this transfer is done using hierarchical structured data (e. g. XML), which reflects the structure of the measured train (see figure 2). On request, he provides the dataset to the rolling stock operator, who in turn can make it available to the rolling stock owners. However, since a measurement data record can contain the data of several rolling stock owners, it must be ensured that a rolling stock owner only has access to data relating to its own wagons. For reasons of confidentiality, it is not allowed to send him the full measurement record, but only the part relating to his wagons. The same applies if the data should be passed to to the manufacturer of specific parts of the train, e. g. breaks. The part manufacturer should only receive data about his parts and not of the parts of competitors. This must be taken into account when developing a concept to ensure the genuineness of the data, so that rolling stock owners are also able to validate the genuineness of the data without having the complete measurement record.

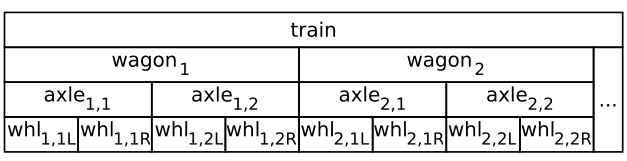

Fig. 2: Example with simplified train/wagon structure

[2] https://poex.io/ – Bitcoin based Proof-of-Existence/Data Integrity Web Service

IV. Research Objectives

A. Requirements

Essentially, the concepts for ensuring data integrity mentioned in section II are based on the fact that the data itself or a unique representation thereof (e. g. a hash, calculated by a cryptographic one-way function) is recorded by a timestamping service. Since every participant of a distributed ledger has access to all information contained in the ledger, it is not suitable for storing confidential information there. For this reason, only the hash of the record to be protected is stored instead of the record itself. However, the original data must still be made available to a verifier so that he can calculate the hash value himself and compare it with the ledger entry. This can be problematic for confidentiality reasons if the verifier is actually not allowed to see all parts of the data. In this case, we need a solution that also supports the transfer of parts of the data without compromising verifiability. We will refer to this as selective disclosure. This requirement for the possibility of selective disclosure can easily be met by hashing the parts of the data individually and applying the existing concepts to ensure data integrity individually. The problem is that at the time the hash is calculated, it is not yet known into which parts the data to be protected will be split or which parts of it will be forwarded to selected parties. For this reason, we need a solution to ensure the genuineness of data record, which would allow verification regardless of *how* the data will be partitioned and shared. In addition, it must be ensured that such a solution does not reveal confidential information to unauthorized parties.

In this paper we investigate to what extent the described approaches for ensuring data integrity on distributed ledgers can provide functionality which ensures confidentiality of the whole data regardless if it is shared complete or in parts. For clarification purposes, we provide an informal definition of the properties we require for dealing with data as described before:

a) Authenticity: It must be verifiable that a data set has actually been recorded and published by the specified sender (e. g. sensor). This requirement also applies to parts of the record.

b) Integrity: It must be verifiable that data received by a party different from the sender (e. g. an infrastructure manager) is equal to the data issued/transmitted by the sender, i. e. that the data was not subsequently modified.

c) Non-repudiability: As soon as a measurement record has been issued, it must no longer be revocable. It must be verifiable that this was issued by the specified sender, even if the sender denies this in case of a dispute.

d) Confidentiality: It must be ensured that the recipient only receives the data for which he is authorized, without compromising the properties of authenticity, integrity and non-repudiability.

B. Research Question

Based on the requirements given in IV-A and the background information regarding the railway sector given in section III, the following research question arises:

> Can we design a system that for selectively disclosed data ensures the integrity, authenticity and non-repudiability while maintaining confidentiality?

We will answer this research question by presenting an example system architecture, which fulfills the requirements mentioned above.

V. Ensuring Genuineness for Selectively Disclosed Confidential Data

Current work uses the concept of Linked Timestamping based on distributed ledger technologies to ensure data integrity. In the following we present an architecture that ensures the confidentiality of data or parts of data without violating the verifiability of data integrity.

A. Design Decisions

The approach we present is intended to work under the following conditions:

a) Working public distributed ledger: We assume that all parties involved have access to a common distributed ledger and that the data contained in this ledger is considered to be tamper-proof.

This assumption is met by most common public distributed ledgers such as Bitcoin, Ethereum and IOTA. Following the example of Karlsson et al., we will refer to this ledger as *support ledger* [13].

b) Public keys are public, private keys remain private: Our approach relies on standard asymmetric cryptography. We assume that the public keys used are globally known. Private keys of the parties remain with the respective party with the following restriction: It cannot be excluded that the sensor manufacturer has access to a copy of the private keys of the sensors (e. g. created during installation). However, it can be assumed that the private key can not be extracted from the sensor by the customer who bought the sensor.

c) Incentive-driven behavior: All parties involved in our concept are assumed to act rationally and incentive-driven. This includes that all types of railway companies mentioned in this concept would follow the protocol presented here during normal operation. Only in the event of incidents some parties might be tempted to tamper with data, since they are incentivized to avert possible consequences like a penalty.

B. Architectural Design

First, the authenticity property requires that it is ensured that data originates from the specified sender. For this purpose we use digital signatures, created by a function $sig(data, signing\_key)$. Each sensor S is equipped with its own certificate/key pair, pub_S and $priv_S$, which is signed by a common authority, e. g. that of the manufacturer. This means that each sensor can be identified individually and it can be ensured that it is actually a product from the manufacturer. The infrastructure manager has a basic trust in the manufacturer and the correct functioning of the sensors, otherwise he would not buy their sensors.

In the following we describe the steps of data collection and transmission from the sensor to the parties listed in figure 1.

a) Step 1: sensor – infrastructure manager: When a train passes by, the sensor creates a data record d that covers observation about the whole train. The sensor now derives a Merkle Tree [12] $H(d)$ according to the hierarchical structure of the record (see figure 3), using a hash function $h(x)$, considering the most detailed elements in the hierarchy as leaves. The hash of each node n in a Merkle Tree is usually calculated by $h_n = h(h_{child_1(n)}, h_{child_2(n)}, ...)$ (we will explain a modification for hash calculation in step 3). After the the root $H_0(d)$ hash of the Merkle Tree has been generated, the sensor creates a digital signature $sig(H_0(d), priv_S)$ of the root hash. The root hash $H_0(d)$ of the Merkle Tree will then be added to the support ledger, and a support ledger transaction reference $txref$, the data record d and the root hash signature $sig(H_0(d), priv_S)$ are sent to the infrastructure manager, either by using a secure network connection or by encrypting the data itself, e. g. using $enc(d, pub_{IM})$.

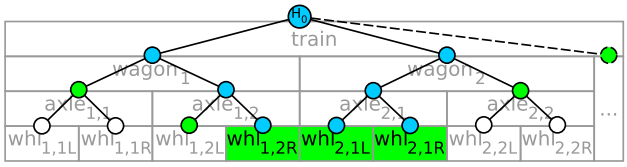

Fig. 3: Example train/wagon structure corresponding to a Merkle Tree

b) Step 2: infrastructure manager – rolling stock operator: The transfer of the complete measurement record from the infrastructure manager to the rolling stock operator is essentially the same as in step 1. In the case of data encryption, the public key pub_{RSOp} of the rolling stock operator $RSOp$ must be used for encryption. The process for verifying the received data on the rolling stock operators' side is identical (computing and comparing the Merkle Tree root hash and verifying the root hash signature).

c) Step 3: rolling stock operator – rolling stock owner: In this scenario only parts of the measurement record should be transferred instead of the entire record, e. g. a few single leaves of the merkle tree.

For this purpose, the confidential parts of the record are replaced by the respective Merkle Tree hashes. If a part of the record covered by a complete branch of the Merkle Tree is not going to be transferred, it is sufficient to transfer the hash of the top node of this area in order to provide integrity verification.

As long as the input value is long enough, it would be infeasible to try to find the original part of the measurement by just iterating all possibilities.

By assigning an individual, random value (*salt*) to each node of the Merkle Tree, we can increase the amount of input values to be tested for a collision with a known hash of a node of the Merkle Tree. The range of the salt value should be large enough to make a possible force attack unfeasible. By using the same size or even larger size compared to the output range of the used hash algorithm, we can decrease the probability of finding a collision to our given hash to the probability defined by the used hash algorithm [14, p. 323f]. It is also important to use different salts per node, otherwise an attacker, which tries to find the original data, already know the most complex part needed for calculating and comparing the respective hash.

The salt must already be generated in step 1 of our architecture. The hash of a node n has to be calculated by $h_n = h(salt_n, h_{child_1(n)}, h_{child_2(n)}, ...)$. For each hash to be calculated on the recipient's side, the respective salt must be included in the transfer (this corresponds to the blue nodes shown in figure 3).

The recipient calculates the hashes for the received parts of the data and then creates the Merkle Tree using the hashes and salts included in the transfer. Authenticity, integrity and non-repudiability can be checked again by checking for the root hash in the support ledger. Confidentially is given since the recipient only receives the data he is authorized to since there is no need for receiving all the data for calculating the Merkle Tree root hash. By using salted hashes it is also not feasible to try to guess untransmitted parts of the data with possibly known structure by using a brute force attack.

VI. Evaluation

A. Setup

For our evaluation (see table I) we were provided with 192 XML files containing realistic measurement values from one train passage per XML file[3]. These file have a size from 3,996 to 177,337 bytes with average size of 18,616 bytes.

For the analysis of the duration of the hash calculation ($h(d)$ and $H(d)$ we ran each test with each file 100 times to increase statistic significance. For both tests, we used the SHA3-256 hashing algorithm.

Our tests also include sending and retrieving the calculated hash to and from a distributed ledger. Since we assume a functional distributed ledger and care about the time the sensor needs to process the data, we do not care about the time it needs for getting our transactions finally confirmed by the network. We chose Bitcoin (Testnet[4]), Ethereum (Ropsten Testnet[5]) and IOTA (Mainnet) for testing this. In all cases we do not operate a full node of the respective ledger locally but use publicly available remote nodes. In case of performance problems using a publicly available remote node (e. g. too many other people using this node), it is possible to improve performance for production use by operating an own ledger node. To increase statistic significance, but without too much spamming the respective ledger, we ran each test with each file 10 times, so 1,920 iterations and therefore new transactions in total per ledger.

[3]Source: voestalpine SIGNALING Siershahn GmbH, https://www.voestalpine.com/signaling/

[4]Alternative Bitcoin, https://en.bitcoin.it/wiki/Testnet

[5]Alternative Ethereum blockchain, https://github.com/ethereum/ropsten

Our test scripts ran on a dedicated Raspberry Pi Model 3B+ to minimize interference from parallel running processes. Since the in-use measurement computers for processing the sensor signals are significantly more powerful as described in section III, our tests with a computer with limited capabilities show a lower limit to the performance of our concept.

B. Evaluation Objectives

The target of our evaluation is to show that a sensor or measuring computer with sufficient computing power is able to publish the calculated Merkle Tree Root Hash on a public Distributed Ledger for each measurement according to our concept. For this reason, we do not measure the CPU time, but the real time, since the frequency of the entries sent to a ledger must be higher than new sensor readings arrive to prevent a growing queue of unsent measurement hashes.

C. Evaluation Results

Table I shows the results of our evaluation. We see that, as expected, the calculation of a Merkle Tree root hash is more complex than the sequential hashing of a file. We also see that the duration of accessing a distributed ledger is much longer than the duration of the hash process, so that the amount of hash time in the total runtime is negligible.

TABLE I: Evaluation calculating hashes including push/pull to Bitcoin, Ethereum and IOTA. $h(d)$ denominates a classical hash calculation; $H(d)$ is a function for calculating the Merkle Tree based on the XML tree.

Task	min	avg	max
Generate $h(d)$	0.42ms	1.67ms	15.16ms
Generate $H(d)$ (unsalted)	7.69ms	35.91ms	351.13ms
Generate $H(d)$ (salted)	7.95ms	39.24ms	411.72ms
Push H_0 to Bitcoin	882ms	1838ms	3610ms
Fetch H_0 from Bitcoin	278ms	701ms	1277ms
Push H_0 to Ethereum	102ms	114ms	782ms
Fetch H_0 from Ethereum	27ms	29ms	35ms
Push H_0 to IOTA	5684ms	9326ms	19844ms
Fetch H_0 from IOTA	5330ms	5407ms	5616ms

D. Merkle Tree Calculation Costs

As part of our work, we have focused on confidentiality as part of ensuring the genuineness of data. One crucial building block of distributed ledger-based integrity solutions are Merkle Trees. With Merkle Trees it is generally possible to verify the integrity of data or parts of data without knowing the complete data covered by the Merkle Tree. For an integrity check, it is sufficient for each part of the data to know either the data or the respective hash, whereas for larger contiguous parts the hash of the corresponding higher-level node in the Merkle Tree is also sufficient.

The literature known to us usually uses binary balanced Merkle Trees to realize a more efficient integrity check compared to sequential hashing of the data in question (see table II). For our approach, we partition the data record into groups following the semantic structure of the measured object (e. g. a wagon, an axle or a wheel, see figure 2). We could also partition it following the internal structure of the data, (e. g. XML, but also JSON etc.). This allows us to easily select parts of the data that we want to transfer and where we only have to transfer the corresponding hash for providing the possibility of an integrity check. The structure of the calculated Merkle Trees therefore corresponds to the structure of the data. However, this structure usually does not correspond to a binary, balanced tree (trains can have more than two wagons, a wagon can have more than two axles, ...), therefore the initial performance gain is lost. This has no practical effect, as the additional time required for building a non-binary, non-balanced Merkle Tree is negligible due to the amount and size of the actual data records (see table III) and the time it needs for publishing the hash to a distributed ledger (see table I).

TABLE II: Computational complexity for doing a hash based integrity check in dependency of record length n

Task	Simple	bin/bal. Merkle Tree
Building Verification Base Hash Time	$O(n)$	$O(n)$
Transfer Hash(es) for Verification	$O(1)$	$O(log(n))$
Verification Time	$O(n)$	$O(log(n))$

E. Fulfillment of Requirements

The requirements of authenticity, integrity, non-repudiability can be fulfilled as shown in the description of the individual transmission steps in V. The confidentiality requirement is also fulfilled, since the values replaced by hashes cannot be restored without further effort (e. g. trying to brute force a hash collision), even for small structured data due to the use of salted hashes. This answers our research question in IV-B, since we are able to give an example architecture which meets our requirements described in IV-A.

F. Using our Genuineness Approach in Real World Railway Systems

Since we only submit the hash of the data to the support ledger and the duration of the hash calculation as described in VI-C is negligible, the actual size of the measured value is negligible for a real world application too. The frequency of measurements, on the other hand, is relevant both for the question of basic feasibility and for transaction costs of the distributed ledger.

Table III shows the average number of train passages measured by various infrastructure companies. Assuming there would be a measurement every 60 seconds (that means a maximum distance between trains of 60 seconds, which is unlikely due to security considerations in train operation), this would result in a maximum of 1,440 entries per day a single sensor has to process.

As we can see in table I, the resource intensive part on our side, namely calculating the hashes, can be done in much shorter period of time than 60 seconds. The push time to a support ledger node barely consumes local computational resources, as it is only waiting for a confirmation from the server for a successful transfer. In the case that multiple hashes need to be pushed to a ledger, it is possible to distribute these requests across multiple servers to avoid a bottleneck

on this side. For this reason, even the unlikely number of 1440 measurements per day poses no challenge from a computational resource point of view. Therefore if the number of measurements per day causes problematic high utilization of computational resources, it would already be limited by train operation safety related regulation.

At the time of writing (April 10, 2019), the average transaction fees of Bitcoin are approx. 2 USD, of Ethereum it is 0.10 USD. Using the idea of OriginStamp [10] of collecting several measurement hashes and publishing a hash of these hashes once per day, costs can be significantly reduced. An alternative would be to use a transaction fee-free support ledger such as IOTA.

TABLE III: Average number and standard deviation σ of train passages per sensor per day from 6 different infrastructure managing companies

Metric	Co. 1	Co. 2	Co. 3	Co. 4	Co. 5	Co. 6
AVG	115.27	76.65	96.33	30.4	41.4	289.42
σ	15.49	4.8	2.28	1.07	1.56	23.99

VII. Generalizing our Approach

Although we have mainly used examples from the field of railway infrastructure, our concept is not limited to this application. Our approach applies to applications where the following goals are supposed to be met:

- Genuineness and confidentiality of data must be ensured once the record has been created.
- It must be possible to share the data or parts of the data without violating the verifiability of genuineness.
- At the time the data set is created, it is not yet known how the data will be partitioned for sharing.

The criteria listed here apply, for example, to the following scenarios:

a) Component Monitoring: Parts of a device (e. g. brakes of a car or train) are monitored by sensors. On the basis of service contracts, e. g. within the scope of warranty claims, the manufacturer of these parts should have access to data relating to his components without seeing data about other components. As long as the integrity of the data is not secured, this would be susceptible to warranty fraud. Passing the complete data recorded within device could reveal sensible information (e. g. home address or frequently visited locations of the car owner). Therefore, from the point of view of the owner of the device, it is desirable to forward only parts of the data, whereas from the point of view of the manufacturer, the verifiability of authenticity should not be compromised.

b) Electronic Health Records: With the digital recording of doctor-patient contacts, data such as diagnosis and type of treatment are recorded. In the case of prescription medication, a patient in Germany receives a prescription in paper form at the end of his visit to the doctor, for which he can obtain the prescribed medication in the pharmacy. Such recipes are relatively easy to forge, as only a stamp and the doctor's signature guarantee authenticity. If the prescribing doctor is not known to the pharmacist and therefore cannot verify his signature, he can only inform himself by telephone about the correctness of the prescription. However, also the doctor on the phone cannot be sure that the caller is actually a pharmacist or that someone wants to illegally obtain confidential information about the patient.

VIII. Conclusion

In our work we have shown that it is possible to ensure the authenticity, integrity and non-repudiability of data while having the possibility of withholding confidential parts of the data without relying on a central Trusted Third Party. We were also able to show that the use of Merkle Trees, even in combination with salted hashes, does not have significant decrease in performance.

With our approach it is possible to produce sensors that prevent manipulation of the data even after distribution over a network of involved parties. This could become an interesting extension of existing concepts for data security, e. g. for securing measurement data in rail wayside monitoring.

References

[1] M. Linden, H. von Kortzfleisch, and M. Arndt, "1c23 railway accident costs: determining the value of preventing a casualty by analyzing national investigation bodies reports (safety-infrastructure)," in *The Proceedings of International Symposium on Seed-up and Service Technology for Railway and Maglev Systems: STECH 2015*. The Japan Society of Mechanical Engineers, 2015, pp. _1C23–1_.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, accessed: 2019-04-12.

[3] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[4] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," in *Advances in Cryptology-CRYPT0' 90*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 437–455.

[5] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.

[6] A. Churyumov, "Byteball: A decentralized system for storage and transfer of value," https://byteball.org/Byteball.pdf, 2016, accessed: 2019-04-12.

[7] S. Popov, "The tangle," https://iota.org/IOTA_Whitepaper.pdf, 2017, accessed: 2019-04-12.

[8] L. Baird, "The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance," 2016.

[9] B. Liu, X. L. Yu, S. Chen, X. Xu, and L. Zhu, "Blockchain Based Data Integrity Service Framework for IoT Data," *Proceedings - 2017 IEEE 24th International Conference on Web Services, ICWS 2017*, no. November, pp. 468–475, 2017.

[10] T. Hepp, A. Schoenhals, C. Gondek, and B. Gipp, "OriginStamp: A blockchain-backed system for decentralized trusted timestamping," *it - Information Technology*, vol. 60, no. 5-6, pp. 273–281, 2018.

[11] C. Madhumohanreddy and G. Raghavendra, "Blockchain-Based Database to Ensure Data Integrity in Cloud Forensics," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology © 2018 IJSRCSEIT*, vol. 4, no. 10, pp. 2456–3307, 2018.

[12] R. C. Merkle, "A Digital Signature Based on a Conventional Encryption Function," in *Advances in Cryptology — CRYPTO '87: Proceedings*, 1988, pp. 369–378.

[13] K. Karlsson, W. Jiang, S. Wicker, D. Adams, E. Ma, R. van Renesse, and H. Weatherspoon, "Vegvisir: A partition-tolerant blockchain for the internet-of-things," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 1150–1158.

[14] A. Menezes, P. van Oorschot, S. Vanstone, S. A. Vanstone, A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, ser. Discrete Mathematics and Its Applications. CRC Press, oct 1996.