Received: 15 May 2019 | Revised: 9 October 2019 | Accepted: 18 November 2019
DOI: 10.1002/cpe.5630

WILEY

SPECIAL ISSUE PAPER

# Trusted systems of records based on Blockchain technology - a prototype for mileage storing in the automotive industry

Katarina Preikschat[1]* | Moritz Böhmecke-Schwafert[1]* | Jan-Paul Buchwald[2] | Carolin Stickel[3]

[1]Department of Economy and Management, Technical University of Berlin, Berlin, Germany
[2]51 nodes GmbH, Stuttgart, Germany
[3]STAR COOPERATION GmbH, Böblingen, Germany

**Correspondence**
Moritz Böhmecke-Schwafert, Department of Economy and Management, Technical University of Berlin, Marchstr. 23, 10587 Berlin, Germany.
Email:
Moritz.Boehmecke-Schwafert@tu-berlin.de

## Summary

Blockchain technology has the potential to bring transparency and trust to a multitude of use cases. Our research demonstrates that the technology can reduce asymmetric information in markets by bridging trust gaps. The combination of blockchain and Internet of Things technology that automatically collects sensor data, provides a feasible, decentralized technological solution for such an inefficient "Market of Lemons" coined by nobel laureate Georg Akerlof. In this paper, we develop a system prototype to reduce mileage fraud on the used car markets. Our work demonstrates the feasibility of a trusted system of records for (vehicle) data such as mileage data using a distributed database based on the public Ethereum network and smart contracts. We have identified eight requirements that are fulfilled by the prototype and the functional logic and design of thesolution can be reproduced to any other application area characterized by a lack of trust between actors or by the absence of a trusted central authority. However, the developed prototype suffers from similar limitations and challenges as the technology itself. Low throughput causes limitations in scalability and transaction costs are unpredictable. Further development of the blockchain technology and considering more cost-efficient consensus mechanisms will address these issues.

KEYWORDS

blockchain, decentralized application, ethereum, internet of things, mileage, system of records, trust

## 1 | INTRODUCTION

In the advent of the Internet of Things (IoT), more and more physical and virtual devices are seamlessly interconnected, and the amount of centrally managed data grows exponentially. User acceptance and adoption of IoT implementations are highly depending on trust and the perception of transparency.[1] The emerging distributed ledger technology of blockchain has the potential to dissolve the centralization of authority in the management of data and facilitate secure sharing of IoT datasets.[2,3] It can provide a trusted system of records for information (eg, transactions) of any kind due to its immutable data and historical traceability.[4] Hence, a trusted and central authority (CA) within a network becomes dispensable.[5] Despite being new and experimental, blockchain technology is increasingly understood as a disruptive and new "General Purpose Technology" being a highly transparent, resilient, and efficient distributed ledger.[6,7] It has the potential to facilitate and solve issues such as information asymmetries that lead to market inefficiencies by replacing trust in human organizations and interactions with trust in the unerring logic of computer-based verification and the power of consensus.[8]

The "*Market for Lemons*" from Nobel prize winner Georg Akerlof is a textbook example in Economics for the consequences of information asymmetries and lack of trusted CAs in markets. Akerlof[9] discussed the used car market that is characterized by fraud because the quality of used cars often cannot be assessed in advance by the buyers. He points out that car vendors have an actual information advantage. Consequently,

[*] Katarina Preikschat and Moritz Böhmecke-Schwafert are designated as first coauthors and have contributed equally to this work.

"*good cars and bad cars*" are sold at the same price as their quality is hardly assessable ex ante.[7(p459)] This hidden information leads to a decrease in market value of high-quality cars, whereas the value of low-quality cars increases. Hence, the incentive to sell high-quality cars is decreasing; thus, the "*bad*" cars drive the "*good*" ones out of the market.[9]

Today, almost 50 years later, information asymmetries such as on the used car market still exists. The value of a used car is often highly influenced by its mileage record along with other factors such as the accident history as it also determines prospective maintenance cycles. Thus, there is an incentive to manipulate the mileage data in order to maximize profit. If we apply Akerlof's[9] model, cars with a manipulated mileage can be considered as "*bad quality*," whereas cars with no manipulated mileage can be defined as "*good quality*." The manipulation of mileage data is considered illegal in most legislations; however, it is neither costly, time-absorbing, nor risky to manipulate the mileage of a used car as it often remains uncovered. There are certain technical approaches currently available to reveal mileage manipulation; however, they often lack accuracy, efficiency, and simplicity as elaborated later on. The prevalence and economical costs of mileage manipulation are highly relevant. Figures on the German market revealed that in total, every third used car is estimated to have an illegally manipulated mileage, which causes more than a 6 billion EUR damage per year.[10]

The 2017 adopted EU regulation (EU) 2017/1151 that came into force in September 2018 aims to facilitate the retrieval of maintenance data from newly produced cars. By this means, it has the potential to reduce mileage fraud as it prompts that car "*manufacturers shall effectively deter reprogramming of the odometer readings*" that records the vehicles total distance from the beginning on.[11] However, this regulation does not solve current issues on the used car market. In order to balance the asymmetric information, a technological solution is required to provide transparency and traceability of vehicle data such as the mileage. Therefore, it needs to be stored immutably and permanently accessible in a trusted system of records while complying to data privacy standards. Moreover, data transmission should be fully automated to decrease the risk of human intervention.

We state that the combination of IoT and blockchain technology provides a feasible decentralized technological solution approach for a "Market of Lemons" such as the used car market characterized by a low-trust environment. In this paper, we develop a blockchain-based prototype to reduce mileage fraud on the used car markets. Our work is intended to be applied to other use cases (eg, leasing companies). Moreover, the developed prototype should demonstrate the general feasibility of a trusted system of records for IoT data based on a public blockchain.

The remainder of this paper is structured as follows. In the following chapter, we introduce the disruptive technology of blockchain as the underlying technology of our prototype as well as the technological background on vehicle mileage storage. Based on this, the third section defines the requirements for an immutable mileage storage system. The fourth section then describes our implementation by elaborating the design decisions and explaining the mechanisms and interfaces of the prototype. The fifth section continues with an evaluation and discussion of the implementation, and the sixth section points out the challenges and limitations. Finally, the seventh section summarizes the paper and highlights implications for future work.

## 2 | BACKGROUND

### 2.1 | Brief introduction to Blockchain

The emerging technology of blockchain[6,12] is often considered as a new "General Purpose Technology". Crowdfunding investments into the technology are skyrocketing and reached $19 billion as of mid-2018,[13] and a numerous amount of use cases exist that go far beyond the financial industry and use cases such as blockchain's[14] most prominent one of "Bitcoin."

Blockchain is defined as a shared distributed ledger that can serve as an irreversible and incorruptible repository of information.[14] It is a digital protocol that is able to record a wide range of items, including asset ownership, asset transactions, and contract agreements in an immutable and secure way.[14] According to Nakamoto,[15] a blockchain network is a peer-to-peer network where all peers (in the following called 'nodes') are equal and no coordinating central authority (CA) exists. Blockchain technology unfolds its potential everywhere where data needs to be managed and verified securely by a trusted third party or intermediary. In addition, it can be considered when multiple individuals and groups are interacting and when the ownership of assets or an asset's characteristics are tracked over time.[16,17]

The concept of smart contracts became a value proposition of a series of blockchain implementations. Already in 1996, long before the advent of blockchain technology, Szabo[18] described the theoretical concept of conditional program code that is executed autonomously. The combination of the core principles of blockchain technology (eg, immutability and consensus) and the smart contracts' deterministic and self-executing business logic written in code allows individuals to interact without trusting each other.[5,19]

A blockchain with smart contracts as business-logic component on the back-end can be extended by a graphical user interface on the front-end that is then called a decentralized application (short: dApp). In contrast to the distributed back-end infrastructure, its application is implemented by a single party or by collectives that deploy a smart contract on the blockchain and build a web application as front-end. This party (or parties) then becomes the smart contract owner and acts as an operating party of the dApp platform that enables the interaction with the blockchain. In particular, the Ethereum ecosystem allows developers to implement dApps that are anchored to the Ethereum blockchain. Ethereum clients serve as a web browser and support a Javascript API object. Hence, web pages can be viewed in the client to interact with the Ethereum Blockchain.[20]

Blockchain technology is coined by two major conflict goals.[8] The first one is a transparency-privacy trade-off. While the core concept of blockchain, an open and transparent ledger that serves to verify ownership of assets, solves the problem of double spending, privacy requirements

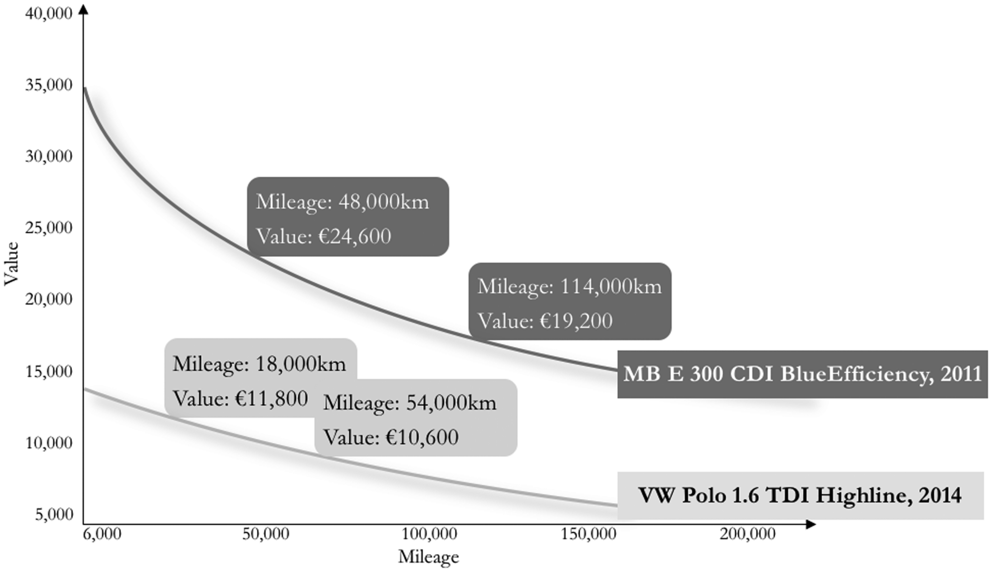


**FIGURE 1** The decrease of a used car's value, when mileage increases. The graphs show that a mileage reduction is more profitable for a higher segment car (eg, Mercedes Benz E 300) in comparison to a lower segment car (eg, VW Polo). Lightly modified according to Brauckmann[27]

of blockchain users have to be fulfilled (this becomes particularly important within blockchain's first use case, the electronic cash system Bitcoin). The second trade-off is a conflict between security and latency. In order to store full history of transactions securely and immutably, a “hash-puzzle” called consensus mechanism has to be solved for every added block to the blockchain. While this increases the resistance to tampering and manipulation, the most prevalent consensus mechanisms are still expensive in time and money. As a result, they slow down the speed of the process when a new block is added to the blockchain.[8]

In response to these two trade-offs, four different types of blockchain evolved, namely public-permissionless, public-permissioned, private-permissionless, and private-permissioned blockchains, which are listed in Table A1.

The first “transparency-privacy conflict” is reflected by the differentiation of reading access and the right to create transactions on the blockchain (compare *x*-axis of Table A1). Thereby, a distinction can be made between a **public** characteristic (high transparency), where every unknown node can contribute to the blockchain with equal rights to read and write transactions, and a **private** characteristic (high privacy), where the ledger is not open for everyone to write or read on it.[8,21] The second “security-speed conflict” is addressed with a differentiation of the access to write blocks on the blockchain and participation in the consensus mechanism, shown on the *y*-axis of Table A1. It is distinguished between a **permissionless** characteristic that gives block writing permissions to everyone, also to untrusted nodes, and a **permissioned** one that gives block writing access only to a limited set of nodes.[21]

A new block of aggregated transactions will only be added to the ledger after the nodes on the network reach consensus about the validity of the transactions.[14] Therefore, an algorithm is used that describes how new transactions have to be processed and how the nodes need to confirm and verify the validity of the transaction before it is added to the longest valid blockchain. That avoids double-spending and provides resistance against denial-of-services attacks.[†][23] The four most common consensus mechanisms that are used in blockchain frameworks such as Ethereum, Bitcoin, or Hyperledger are Proof of Work,[24] Proof of Stake,[19] Proof of Activity,[25] and Byzantine Fault Tolerance.[19] Their properties can be distinguished with respect to latency, throughput, energy consumption, and scalability, and they suggest different application scenarios as illustrated in other works.[6,17,24-26]

## 2.2 | Mileage manipulation in used car markets

The economic costs of mileage manipulations are high and are driven by several incentives such as to illegitimately claim warranty services, decrease leasing rates or we focused on before, and exploit information asymmetries on the used car market, where the mileage of a vehicle is an elastic price determinant.[10] Figure 1 depicts the development of the resale prices of a Mercedes-Benz class E (MB E) and Volkswagen (VW) Polo exemplarily. If the mileage of an MB E is reduced by 66 000 km, the value will increase by €5400. If the mileage of a VW Polo is reduced by 36 000 km, the value will increase by €1200.[27]

According to estimations of the German automobile club ADAC, the market price of a manipulated car is on average €3000 higher than its value.[10] In addition, buyers of cars with manipulated mileage data cannot comply to maintenance schedules, and unexpected repairs increase the economic loss further. Moreover, in accordance to Akerlof's[9] theory, insecurities with respect to the quality (eg, the reliability of mileage data) of used cars influence the purchasing decisions and thus lead to further market inefficiencies.

The locations and means of mileage documentation in a car are kept in secret by the original equipment manufacturers (OEM). The topic is widely discussed in media, and it is assumed that the mileage data is stored in many different places in the car, eg, in the instrument cluster,

[†] Denial-of-Service (DoS) attacks occur if the server is overloaded due to a huge amount of requests caused by hackers.[22]

engine control unit, and many other different ECUs (electronic control units). The mileage data visible to the user is displayed in the tachometer of the instrument cluster and can be manipulated easily by using on-board-diagnosis dongles of the second generation (OBDII). Originally, OBDII was introduced to read data and error codes of certain ECUs in a car in order to facilitate maintenance. A socket for an OBDII dongle is required for all cars produced after 1996 in the US and 2001 in the EU.[28]

It is difficult to reveal mileage fraud. A comparison of all ECUs would be necessary to exclude any mileage manipulation and fraud. There are smartphone applications such as "Carly" that allow to read out the mileage by means of OBDII dongles from certain ECUs.[29] However, the OBDII dongles themselves are often subject to manipulation. Another possibility is the professional estimation of the mileage through experts who analyze the wear and tear of the interior and other car components such as the engine.[30] This process is very complex though, and precision is not guaranteed. Last but not thr least, hardware security modules (HSM) exist, which encrypt and store the mileage on a separated module next to the ECUs similar to a black box.[10] This storage is considered as immutable and secure, but there is neither an interface to the user nor a transmission to a database. Consequently, a trusted system of records that is transparent and easily accessible for the users is necessary. In the following, we describe the requirements for such a system to store sensor data such as the mileage of a vehicle in a distributed database.

## 3 | REQUIREMENTS FOR A TRUSTED MILEAGE STORAGE SYSTEM

Currently, there is no technical solution to view the correct and immutable mileage history data of a vehicle in a transparent trusted system. We propose a system that is based on blockchain technology to overcome the problem of asymmetric information on the used car market. Blockchain technology eliminates the need for a trusted third party (eg, a certification authority) by providing an immutable and timestamped distributed system of records.[4,16] These attributes make it superior to central database approaches.[31] However, the development and maintenance of the decentralized application, particularly the IoT hardware device and software setup,has to be initialized by an independent operating (third) party in order to ensure the reliability of the generated data (compare with[32]). This does not affect the advantages of immutability of stored data. The operating party is ideally a car manufacturer, a regulatory institution or a consortium of those development and maintenance of the decentralized application has to be initialized by an operating (third) party. This could be for example the OEM or certain regulatory institutions. The general requirements for a secure and tamper proof mileage storage system are summarized in Table A2. In the following, we will briefly discuss the requirements that provide the basis for our prototype and evaluate their fulfilment in Section 5.

A potential buyer of a used car should be able to retrace the development of the mileage and reveal potential manipulations with the help of the prototype. A car vendor should be able to use the prototype for signaling in order to increase the authenticity of their car's mileage specifications. As discussed in the background section, existing approaches by means of CAs are insufficient. Moreover, the entire used car market should be covered by the prototype so that every car should be able to send and store mileage data safely. Reading a vehicle's mileage on the prototype should be free of charge, whereas the transaction costs of registering a vehicle should be covered by the vehicle's owners that could convert the signaling of the mileage registration into a price premium. The acquiring party should be able to read the mileage history before the purchase decision. We summarized these aspects in Requirement 1 and 2 (R1 and R2 in Table A2). The process of writing and reading mileage data should be as intuitive as possible; hence, the prototype and its user interface need to be handled simply (R3).

As Lemieux[32] points out, a blockchain-based system of records often faces the major challenge in ensuring the correctness and reliability of data entries. If the data can be manipulated before storing it on the blockchain, a blockchain-based system cannot be trustful. Therefore, the proneness to manipulation has to be ensured at the level of the data sources (R4). In addition, the data creation timestamp that facilitates the backtracking of mileage data has to be correctly and immutably attached to the respective transaction. Human interactions with the prototype need to be at the lowest possible level to enhance the security of the prototype because they are carrying the highest risk of manipulation and fraud. Therefore, a prototype needs to seek for a complete automatic process for the tracking of mileage data and timestamping of transactions (R5).

Another requirement is the regular frequency of tracking and storing the mileage data. The mileage should be only stored in the blockchain in a frequency where the value of the car changes (R6).

Moreover, the mileage data should always be assignable to only one vehicle. The unique Vehicle Information Number (VIN) could be used to identify the specific car and to check the mileage history (R7). Finally, yet importantly, the requirements of the General Data Protection Regulation (GDPR) and the anonymity of users have to be considered (R8). The owner of a vehicle should be the sovereign of their own data and provide consent for mileage history tracking. In addition, all users of the system should have the right to stay anonymous. Reading out the mileage history should only be possible with the VIN that the vehicle vendor provides with consent to the potential buyer in order to check the mileage of the vehicle (R8).

## 4 | PROPOSED IMPLEMENTATION

In this section, we present the design decisions of the blockchain-based solution, the systems' architecture, and the realization of the prototype on a test node. We describe and document the implementation in detail in order to facilitate its reproduction to similar use cases that seek to establish a secure, tamper-proof, and easily accessible system of records.

## 4.1 | Design decisions

We consider an implementation on an ECU level as an essential prerequisite for a potential industry standard of a blockchain-based mileage storage systems because ECUs are immutable for users and can only be modified by the manufacturer. An implementation on an OBDII socket to store and transmit mileage data would not be sufficient because they are prone to manipulation (see above) and OBDII dongles are used for mileage manipulation. Hence, it is recommended to retrieve mileage data directly from ECUs. Moreover, an ECU fulfills the technical requirements to run a blockchain client; thus, the mileage data can be regularly transmitted and stored on a blockchain.

For this paper, we have used an additional device to simulate an ECU black box because we did not have a cost-effective access to an ECU. However, we want to highlight that a reproduction directly on an ECU is necessary for a market solution and would not cost much effort. Therefore, we use an additional device,[‡] a tool called "FlexDevice" that simulates an ECU black box (it is placed inside the vehicle and can additionally be connected to an ECU) and tracks as well as transmits the mileage data to a blockchain.

We extend our black box setup with a Raspberry Pi that Raspbian with existing blockchain clients as we did not find blockchain clients for the operating systems that are specific in the automotive industry and we wanted to simplify the setup to prove general feasibility. We describe the Raspberry Pi as Mileage2Blockchain module in our prototype setup because it acts as a gateway between the black box (the "Flex Device") and a blockchain. The black box can transfer logged data via a transport protocol to the Raspberry Pi, on which a blockchain client creates transactions with mileage data to the blockchain.

The blockchain client only creates transactions every 1000 km in order to avoid unnecessary transactions as we assume that a vehicle loses significant value within this interval (see background section). Mileage data is generated by another "FlexDevice" (see above) instead of using an actual car. Using this virtual data creator saves costs and time as the second FlexDevice can, without any effort, generate and modify mileage data during the development of the prototype.

As described in the previous section, the creation of a timestamp is required as well to certify the data generation time. As the black box that simulates the data is not capable of creating timestamps (with an ECU, it would be possible though), a timestamp is created by the Raspberry Pi to certify the data when a transaction is made and sent to the blockchain. The retrieval of mileage data for users should be as simple as possible without any knowledge of the back-end technology of blockchain. Thus, we developed a web application that is accessible from any browser and further described in Section 4.4.

An important design decision is the selection of an appropriate type of blockchain. In the following, we discuss and evaluate four blockchain characteristics.

The blockchain should be open for all actors on the used car market. Therefore, we consider a public blockchain. The network cannot be limited to only known and trusted users. Furthermore, every actor should have the possibility to write transactions, ie, to enable their vehicle to write the current mileage regularly into the blockchain. In addition, all actors should be able to read all mileage entries within a user interface. Hence, neither the access to write transactions nor the access to read transactions is restricted, which corresponds to the characteristics of a public blockchain (see Section 2.1). Moreover, the consensus mechanism should be open for every node of the network to ensure a trustful system without any CA or trusted third party. Traditional central databases that are administered by a CA have higher costs and are less secure because of many intermediary actors.[16] In addition, a network supervised by a CA does not necessarily guarantee immutability. This indicates that the network needs to be permissionless. Consequently, we use a public-permissionless blockchain for the prototype (compare Table A1).

We use the generic public-permissionless Ethereum platform for the mileage prototype as it allows users to build their own applications on top of a blockchain stack. Ethereum is often used when complex and repetitive functionalities are required and a huge amount of transactions is expected,[30] similar to the amount of data generated if all vehicles in Germany were equipped with an automated mileage storage system (ie, under the assumption that all 45.80 million cars in 2017[33] store their mileage data regularly). In addition, the Ethereum script is Turing-complete; thus, it enables the execution of any desired programming logic on the blockchain and provides the possibility of smart contracts. This is a crucial prerequisite to execute and monitor mileage-storing processes after a change in ownership. Ethereum currently uses the consensus mechanism proof-of-work (PoW) where the consensus depends on the computational power of the so-called "mining nodes" (sometimes, on memory or user interventions as well) and hence has high time and resource consumption.[19,21] Other nodes act as referees and decide on the block's validity[8]; thus, PoW is considered as highly secure, tamper proof, and immutable.[25] Bitcoin as a platform drops out as an implementation because among other reasons, it is not Turing-complete (limited functionality with respect to smart contracts) and moreover requires more computation time and transaction costs than Ethereum.[25]

In addition, the Ethereum platform unifies a high number of available client software and libraries for different platforms and programming languages. Furthermore, the development community of Ethereum is large and provides well-developed and intuitive frameworks, libraries, and testing environments. With these environments, which are called test networks, the prototype can be implemented and tested easily, privately, and without paying real transaction costs.[34] In general, transaction costs in the Ethereum network are measured in consumed computational power, called "gas," and not directly in the cryptocurrency Ether. The total amount of consumed gas for one transaction is the "gas usage." When a transaction is executed, the sender has to specify the so-called "gas limit" they are willing to provide for the completion of the transaction. In

[‡] The "FlexDevice" of the company "STAR COOPERATION GmbH" provides an electronic platform for the automotive environment with different application scenarios (eg, simulation of car data). Because of its many interfaces, it can be used for rapid prototyping (see the appendix A.2).

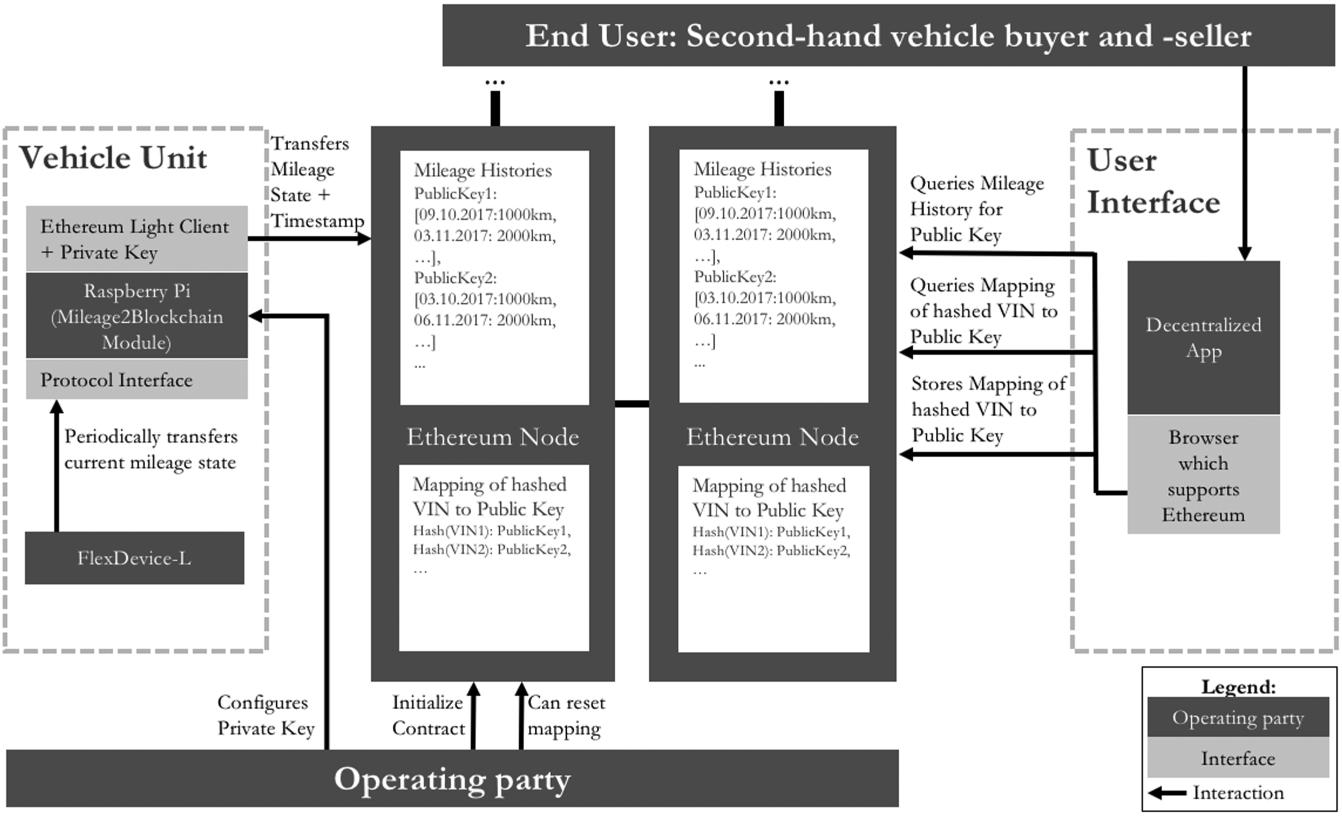


**FIGURE 2** The Mileage2Blockchain prototype system architecture: First of all, the operating party configures the private and public keys of the vehicle units and initiates the smart contracts. The vehicle unit is then able to transfer the mileage data and timestamp to the next Ethereum full node. The mileage data is then stored for the public key of the vehicle unit. In order to assign mileage data to a car, the users can register their car with the VIN by using a web application. This registration can be deleted by the operating party, but the mileage data mapped to the public key remains immutable on the Ethereum blockchain

addition, the sender has to determine a "gas price," the price per unit of gas in Ether. This incentivizes a miner to process the sender's transaction. If the gas price is too low, it is possible that the miner ignores the transaction.[34,35] The exact value of the gas price in the prototype does not matter because no miners need to be incentivized within the development test net; thus, every transaction is confirmed immediately. Based on this determination of general design decisions, the component and system architecture are designed in the following.

## 4.2 | Component design and system architecture

Four functionalities are defined that have to be handled by the business logic within the smart contract of the prototype. These functionalities are summarized in Table A3. They determine the prototype's architecture that is further illustrated in Figure 2.

The architecture of the prototype includes all involved parties and their interactions. The first party is the operating party, which is responsible for initializing the accounts, ie, the private and public keys of each car. While the private key is stored in the light client of the Raspberry Pi (in a scaled application, it would be the ECU), the public key is transparently available for the car owner (eg, printed on the vehicle module) and used as a public identifier of the vehicle in the blockchain network. The operating party also initializes the smart contract on the Ethereum blockchain; thus, it becomes the smart contract owner.

The vehicle unit consists of the black box for generating data and data tracking (the two "FlexDevices") and a Raspberry Pi (Mileage2Blockchain module) that includes the Ethereum light client. The light client connects to an Ethereum node to transfer the mileage and the corresponding timestamp to the Ethereum blockchain, where it is stored with its related public key in a first storage. This storing function is realized through the smart contract function "*storeMileage (mileage, timestamp).*" In order to create gapless mileage histories, the data are stored from the beginning when the Ethereum client is implemented in the vehicle.

The assignment of mileage data to a car using a public key is considered unwieldy. Therefore, a second storage of a mapping of the VIN to the public key is created on the blockchain. In addition, only the hash value of the VIN and not the plain text is stored to ensure data protection. The storage is fed through a user registration process executed by the car owner or a supporting third party who gives consent with the registration for the allocation of mileage data to the VIN. The smart contract function called "*mapVinToPublicKey (hashedVin, publicKey)*" is used.

In case of an erroneous registration or termination of the assignability of mileage data to a vehicle, there needs to be a function to eliminate the mapping of the VIN to a public key by the operating party (ie, the smart contract owner). However, only the mapping of the hashed VIN to

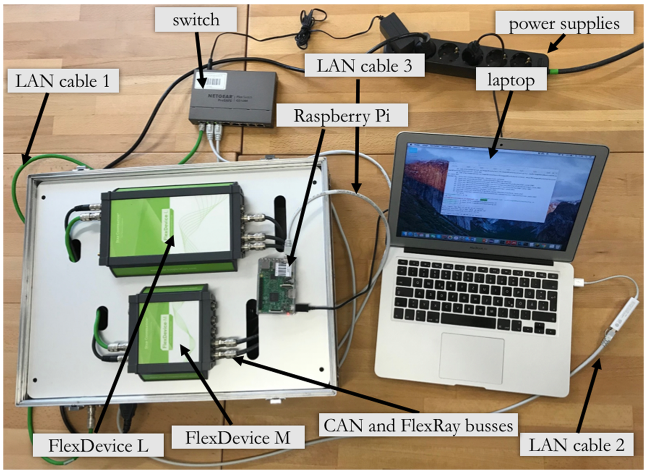


**FIGURE 3** Mileage storing via blockchain prototype hardware setup

the corresponding public key (which is the hashed VIN itself) will be overwritten, the data that is stored on the Ethereum blockchain under the public key remains immutable. This is done through a smart contract function called *resetPublicKeyMapping (hashedVin)*.

Users (eg, vehicle buyers) can interact with the blockchain via a user interface, respectively, a web application in a browser. They can query the mileage history for a VIN via the web application. The public key is queried for the hash of the entered VIN in order to retrieve the mileage history for the public key in the back-end. Therefore, two further functions are included in the smart contract. The function "*getPublicKey (hashedVin)*" returns the public key to a given hashed VIN, and the function "*getMileage (publicKey)*" returns the mileage history for a given public key. These functions are not essential for the functionality of the smart contract because the mileage data is retrievable through a direct request accessing public variables in the contract. Therefore, they do not need to be implemented. The registration process is executable via the web application as well.

## 4.3 | Description of hardware and software setup

In this section, we describe the system architecture which focuses on fast prototype development within the private test node ethereumjs-testrpc, executed with the framework Truffle. All scripts running on the Raspberry Pi are written in Java. The prototype is realized by means of a MacBook Air OS X El Capitan. In the following, the used software and hardware are described first. Subsequently, the programming of the smart contract is explained.

Figure 3 illustrates the hardware setup of the closed network. The data creator, the "FlexDevice M" (a former version of the "FlexDevice L" with less functionalities), simulates mileage data and sends it via communication busses to the black box, the "FlexDevice L," which tracks and sends mileage data via Ethernet (LAN cable 1) and via a network switch to the Raspberry Pi. The mileage is screened toward a change of at least 1000 km in comparison to the previous value stored on the blockchain; then, a timestamp is created, and both are sent via LAN cable 3/2 to the blockchain and stored there. This is explained further in the following. In addition, Figure 4 shows this setup more detailedly, including all software units that are discussed in the following as well. Moreover, the technical specifications of the hardware setup (data generator and black box) are listed in Appendix A.2. In the following, we introduce these components, along their role and functionality in the prototype setup.

### 4.3.1 | Running the Blockchain

Due to the early-stage development, a private test net must be set up first. Therefore, we chose the ethereumjs-testrpc Ganache-CLI that is characterized by fast processing.[6] With an ethereumjs-testrpc client, a full node with Geth-client-behavior is simulated locally. Therefore, it is no real test net but rather a single test node[36] that runs on a laptop within the set up (compare Figure 3 and Figure 4) in order to avoid transaction costs for this prototype (these costs are discussed in Sections 5.2 and 6.3). Using a test node in the prototype has the advantage that transactions are confirmed immediately because no consensus mechanism exists, and in addition, accounts can be equipped with an unlimited amount of Ether. Within the simulated network, two accounts that are each holding one pair of keys are determined with an account balance of 100 Ether. The first account is assigned to the "vehicle seller." The second one is allocated to the corresponding vehicle and stored on the Mileage2Blockchain module.

### 4.3.2 | Smart contract interaction

The smart contract needs to be developed first, then compiled, and finally deployed on the blockchain. Thus, an own address in the network is assigned to the smart contract. Consequently, it can receive transactions for executing the functions. To simplify the development on Ethereum, the environment and testing framework Truffle is used (compare Figure 4). Truffle initiates the project directory structure and enables built-in smart contract compilation, linking and deployment, binary management, automated contract testing, scriptable deployment, and a migrations framework.[37] In order to interact with the smart contract in a browser, a client-side JavaScript abstraction needs to communicate with the smart contract.[38] Hence, the software web3.js library is used to connect the application's front-end (the GUI) with its back-end (the blockchain or full

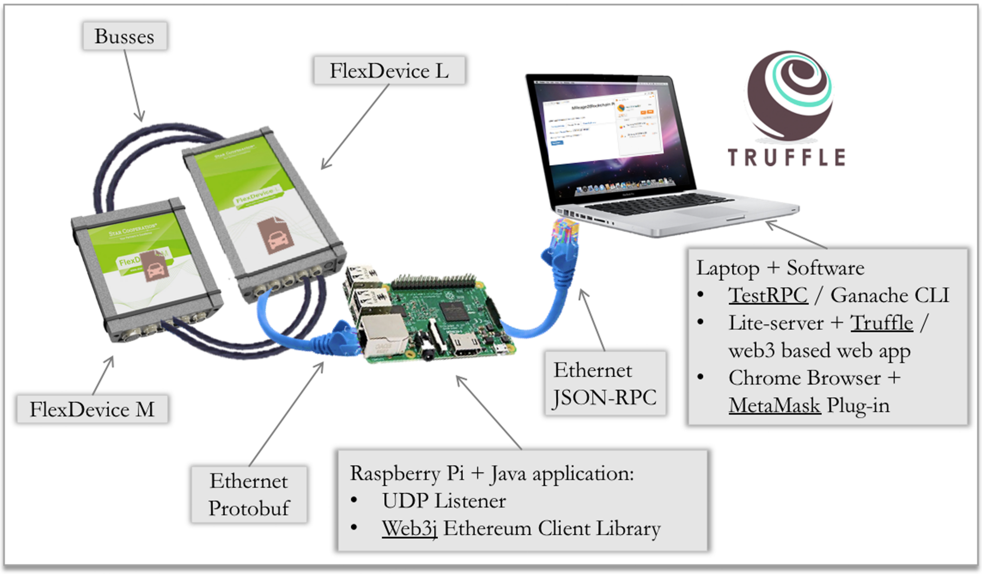

Busses
FlexDevice L
TRUFFLE
Laptop + Software
• TestRPC / Ganache CLI
• Lite-server + Truffle / web3 based web app
• Chrome Browser + MetaMask Plug-in
Ethernet JSON-RPC
FlexDevice M
Ethernet Protobuf
Raspberry Pi + Java application:
• UDP Listener
• Web3j Ethereum Client Library

**FIGURE 4** The prototype setup, including software components

node). This library is available via Node Package Manager (npm). In the web3.js library, there are also so-called JSON-RPC objects. These objects are used to transfer data from the client on the Raspberry Pi to the blockchain.[39] In the prototype, we used Truffle to automatically create the required stubs to access the smart contract.[38] With that, the smart contract functions are directly available and useable by JavaScript code in the web application.

### 4.3.3 | Data creation

Vehicle data similar to data generated by an ECU (ie, sensor data of a vehicle) is simulated by the older FlexDevice M. The mileage progress is predefined in a way that demonstrates an increasing mileage progress, including manipulation events. Because one of the communication busses between the FlexDevices, called the FlexRay bus, has a latency of at least 5 milliseconds, mileage data is created every 5 milliseconds. In order to ensure a traceable data progress suitable for demonstrations, the mileage increases by 1000 km per 10 seconds. Consequently, the mileage increases by 500 m per 5 milliseconds. In order to illustrate manipulations as well, the mileage decreases every 4000 km (that means every 40 seconds) by 2000 km. In contrast, the timestamp is created by the Raspberry Pi (system time).

### 4.3.4 | Data tracking

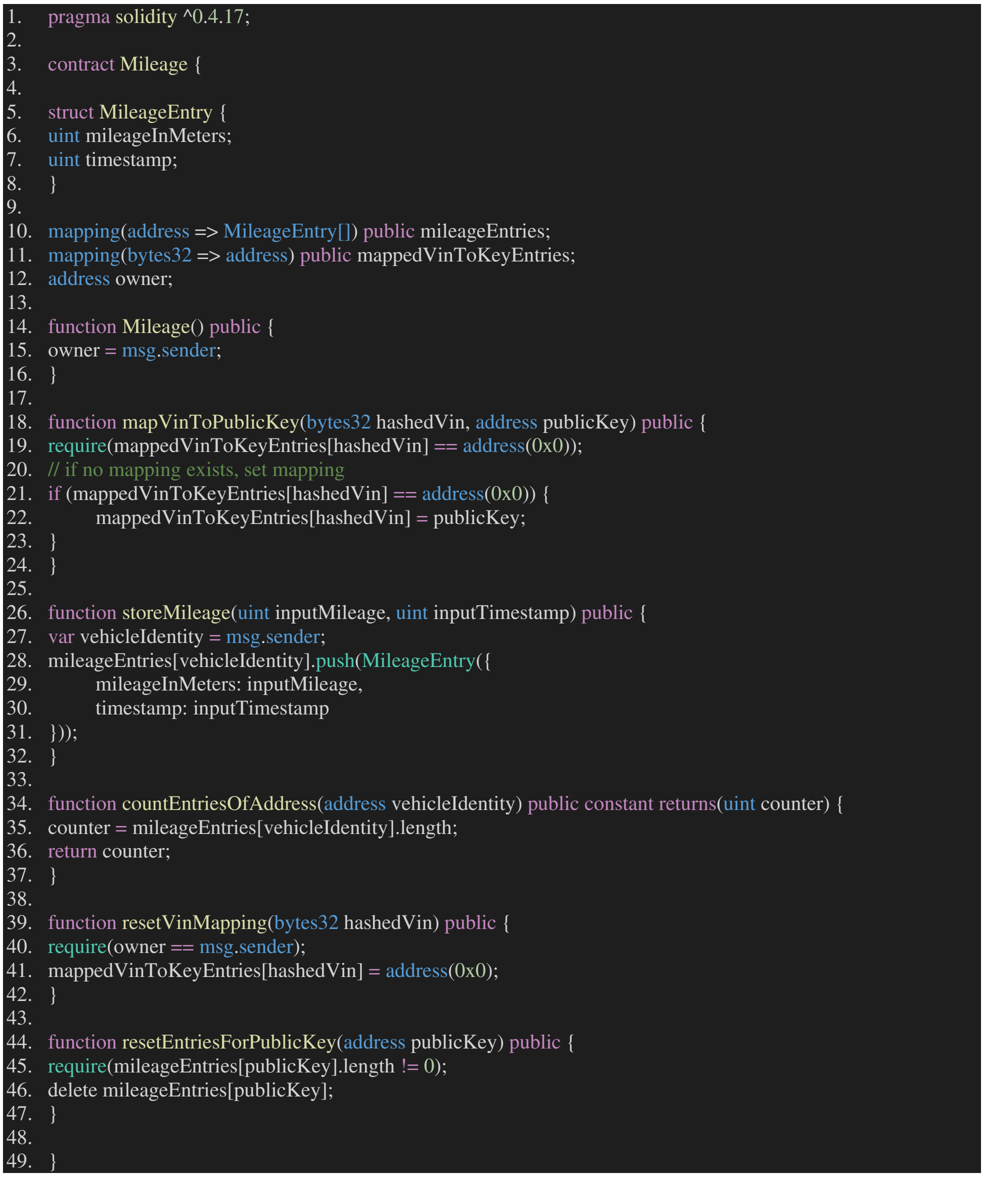
After mileage data is created, it is transferred via FlexRay and CAN (controlled area network) busses to a second development tool, the previously mentioned black box, which is located between these busses (compare Figure 4). It tracks all car data and serializes it via the protocol buffer protocol of Google. This way, the data is packed in a data format that is easily transferable via a transport protocol. Next, the data is transferred through an UDP protocol via Ethernet to the Raspberry Pi, where the data packages are deserialized (unpacked) by a Java class called "Mileage2BlockchainListener."

### 4.3.5 | Data storing

As a next step, the Java program on the Raspberry Pi checks whether the mileage has changed by at least 1000 km compared to the last recorded entry (or zero in the beginning). If it does, the data is sent to the blockchain through a light client on the Mileage2Blockchain module (ie, the Raspberry Pi). The Ethereum light client is realized through the simple usage of the lightweight Java web3j library. Web3j is an Ethereum client library for Java and facilitates the interaction with the smart contract via native Java code using a configurable account with a private and public key and a contract address.[40] In order to connect to an Ethereum full node, the Mileage2Blockchain module also includes the node's URL in its configuration. The module determines the current timestamp and calls the smart contract function *storeMileage (mileage, timestamp)*. This function is then transformed by the web3j library into a JSON-RPC call addressing the smart contract and being sent to the configured full node (compare Figure 4). The identity of the vehicle is determined from the sender address of the transaction, which corresponds to the light client's public key. The smart contract then creates a new mileage history record for the public key using the given mileage and timestamp values.

### 4.3.6 | Running the decentralized app

In order to make the blockchain data accessible for as many users as possible, a web application is developed, which interacts with the blockchain via web3.js. For the prototype, the browser extension MetaMask is used to inject web3.js (compare Figure 4). MetaMask provides an Ethereum light client directly in a Google Chrome browser (also in Firefox and Opera) and enables managing identities (ie, private and public keys) and transactions via an interface. MetaMask is also a wallet and allows the user to view their account balance as well as their transaction history.[41] In



```
1.  pragma solidity ^0.4.17;
2.
3.  contract Mileage {
4.
5.  struct MileageEntry {
6.  uint mileageInMeters;
7.  uint timestamp;
8.  }
9.
10. mapping(address => MileageEntry[]) public mileageEntries;
11. mapping(bytes32 => address) public mappedVinToKeyEntries;
12. address owner;
13.
14. function Mileage() public {
15. owner = msg.sender;
16. }
17.
18. function mapVinToPublicKey(bytes32 hashedVin, address publicKey) public {
19. require(mappedVinToKeyEntries[hashedVin] == address(0x0));
20. // if no mapping exists, set mapping
21. if (mappedVinToKeyEntries[hashedVin] == address(0x0)) {
22.         mappedVinToKeyEntries[hashedVin] = publicKey;
23. }
24. }
25.
26. function storeMileage(uint inputMileage, uint inputTimestamp) public {
27. var vehicleIdentity = msg.sender;
28. mileageEntries[vehicleIdentity].push(MileageEntry({
29.         mileageInMeters: inputMileage,
30.         timestamp: inputTimestamp
31. }));
32. }
33.
34. function countEntriesOfAddress(address vehicleIdentity) public constant returns(uint counter) {
35. counter = mileageEntries[vehicleIdentity].length;
36. return counter;
37. }
38.
39. function resetVinMapping(bytes32 hashedVin) public {
40. require(owner == msg.sender);
41. mappedVinToKeyEntries[hashedVin] = address(0x0);
42. }
43.
44. function resetEntriesForPublicKey(address publicKey) public {
45. require(mileageEntries[publicKey].length != 0);
46. delete mileageEntries[publicKey];
47. }
48.
49. }
```

**FIGURE 5** Source code of the smart contract (Mileage.sol)

the prototype setup, the web application runs on a "lite-server" (a local web server). It provides static files in order to serve the front-end. In the following, the programming code of the smart contract is described and discussed.

### 4.3.7 | Programming smart contracts

The source code of the smart contract is depicted in Figure 5 and explained in the following. First of all, a new data type called *struct* is created in order to describe data, which is transferred from the vehicle unit to the blockchain and stored there. Therefore, the *struct MileageEntry* is built, which consists of the objects *mileageInMeters* and *timestamp*, both of the data type "uint" (unsigned integer) (compare Figure 5, lines 5 to 8). This *struct* is generic and expandable arbitrarily with other variables of vehicle data.

In addition, two mappings are created: one of a hashed VIN to a public key and one of a public key to a *MileageEntry*. Hence, a mapping consists always of a key and a value. If the key is queried, the value is provided as defined in lines 10 and 11 of Figure 5. Consequently, if the hashed VIN (type: bytes32[§]) is queried, its related public key (type: address[¶]) is provided. Moreover, when the public key (type: address) is queried, all relating mileage entries (type: *MileageEntry[]*), including a mileage and timestamp in an array, are given. The following lines are comprised of several functions. Lines 18 to 24 (*mapVinToPublicKey (bytes32 hashedVin, address publicKey))* first check whether a mapping of a hashed VIN to a public key exists (ie, its value is not equal to the default value address(0x0)[#] ). If it does, it will throw an error which is expressed in the programming language Solidity through the predefined function *require()* in line 19. If a mapping does not exist, it will map the hashed VIN to the public key.

[§] "bytes32" is a data type in Solidity which holds a value of fix length of 32 bytes.

[¶] "address" is also a data type in Solidity and holds a value of fixed length of 20 bytes.

[#] "address(0x0)" is the default value and stands for "0x0000000000000000000000000000000000000000."

The function storeMileage (uint inputMileage, uint inputTimestamp) between lines 26 and 32 describes the adding of a new MileageEntry, allocating it with the public key of the transaction sender, including the *mileageInMeters* and a timestamp to the array *mileageEntries*. As a result, the mileage history for the sender's (ie, the vehicle) public key in the blockchain is extended by a new mileage and timestamp entry.

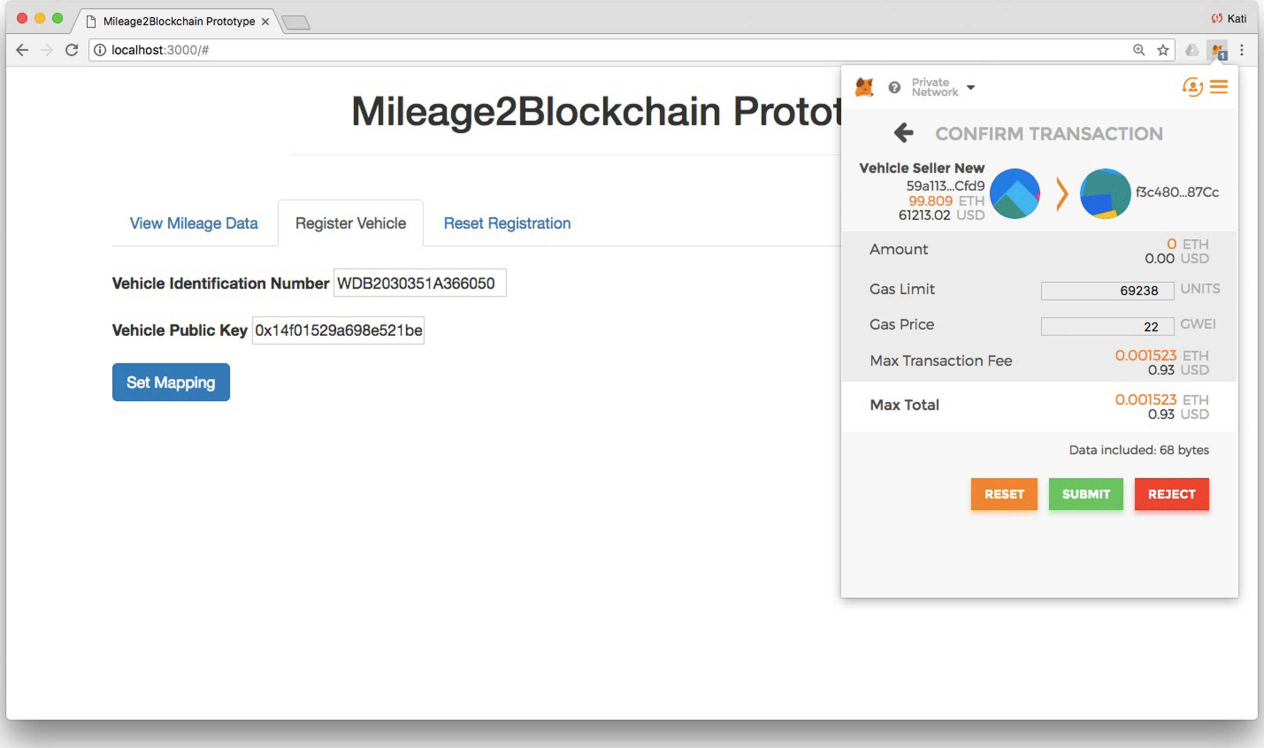
The function *resetVinMapping (bytes32 hashedVin)* in lines 39 to 42 describes that only the owner of the smart contract (ie, the operating party) can reset a mapping of a public key to a hashed VIN by setting its public key to the default value *address(0x0)*. An additional function is necessary to query all mileage entries for a vehicle (VIN) using the corresponding public key. The function *countEntriesOfAddress (address vehicleIdentity)* in lines 34 to 37 counts all mileage entries for a VIN respectively a vehicle's public key.

The last two functions are added and implemented in order to avoid bugs during the Ganache-CLI and MetaMask application. Tests revealed that Ganache-CLI does not work correctly if it is called on the same machine several times with the same accounts. Therefore, the function *resetEntriesForPublicKey (address publicKey)* (lines 44 to 47) supports and resets all entries to a public key in order to execute the demonstration with the same parameters again. This function only exists for the demonstration setup and would not be necessary in the public Ethereum network.

## 4.4 | Functional flow and user interface of the prototype

The first "FlexDevice," the data generator, starts to simulate different vehicle data such as mileage data that increases by 500 m every 5 milliseconds. All data is then tracked and structured by the second "FlexDevice" that serves as a black box and subsequently sends the data to the Raspberry Pi via Ethernet and protocol buffer. In the next step, the Mileage2Blockchain module deserializes the packages and sorts the data by their header to select only the packages that carry mileage data. The Ethereum client on the Raspberry Pi compares whether the unpacked mileage differs 1000 km or more from the previously stored mileage status and if so creates a timestamp and sends it together with the mileage value as a transaction to the blockchain on the local test node Ganache-CLI. The data is signed by the client's private key and stored as transaction from the sender's public key to the public key of the smart contract, which was deployed on the blockchain. Using only a private test network consisting of only one node the transaction is confirmed immediately and mined in a new block. A user interface has been developed to facilitate the interaction between users and the blockchain in the dApp. In this web application, a vehicle has to be registered by the owner or the operating party if its mileage history should be retrievable by the public. After entering the socalled Mileage2Blockchain prototype web page (see Figure 6), the user types in the registration details (ie, the VIN and public key), and a MetaMask notification opens automatically in the web browser (compare Figure 6 on the right).

The MetaMask plugin proposes an optimal gas limit and gas price and calculates the expected maximum transaction fee for the registration process. The user can modify these values before confirming the transaction. If the specified amount of gas meets or exceeds the expectations of



Vehicle Identification Number WDB2030351A366050

Vehicle Public Key 0x14f01529a698e521be

Vehicle Seller New
59a113...Cfd9
99.809 ETH
61213.02 USD
f3c480...87Cc

Amount 0 ETH 0.00 USD

Gas Limit 69238 UNITS

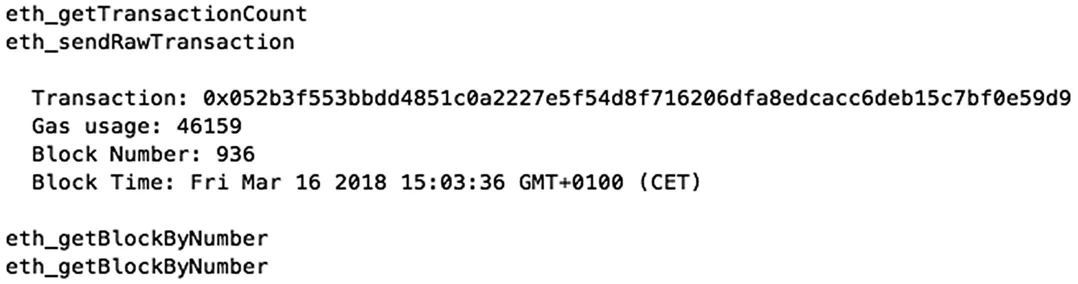
Gas Price 22 GWEI

Max Transaction Fee 0.001523 ETH 0.93 USD

Max Total 0.001523 ETH 0.93 USD

Data included: 68 bytes

**FIGURE 6** Vehicle registration via the web application



```
eth_getTransactionCount
eth_sendRawTransaction

  Transaction: 0x052b3f553bbdd4851c0a2227e5f54d8f716206dfa8edcacc6deb15c7bf0e59d9
  Gas usage: 46159
  Block Number: 936
  Block Time: Fri Mar 16 2018 15:03:36 GMT+0100 (CET)

eth_getBlockByNumber
eth_getBlockByNumber
```

**FIGURE 7** The registration transaction is listed in the blockchain

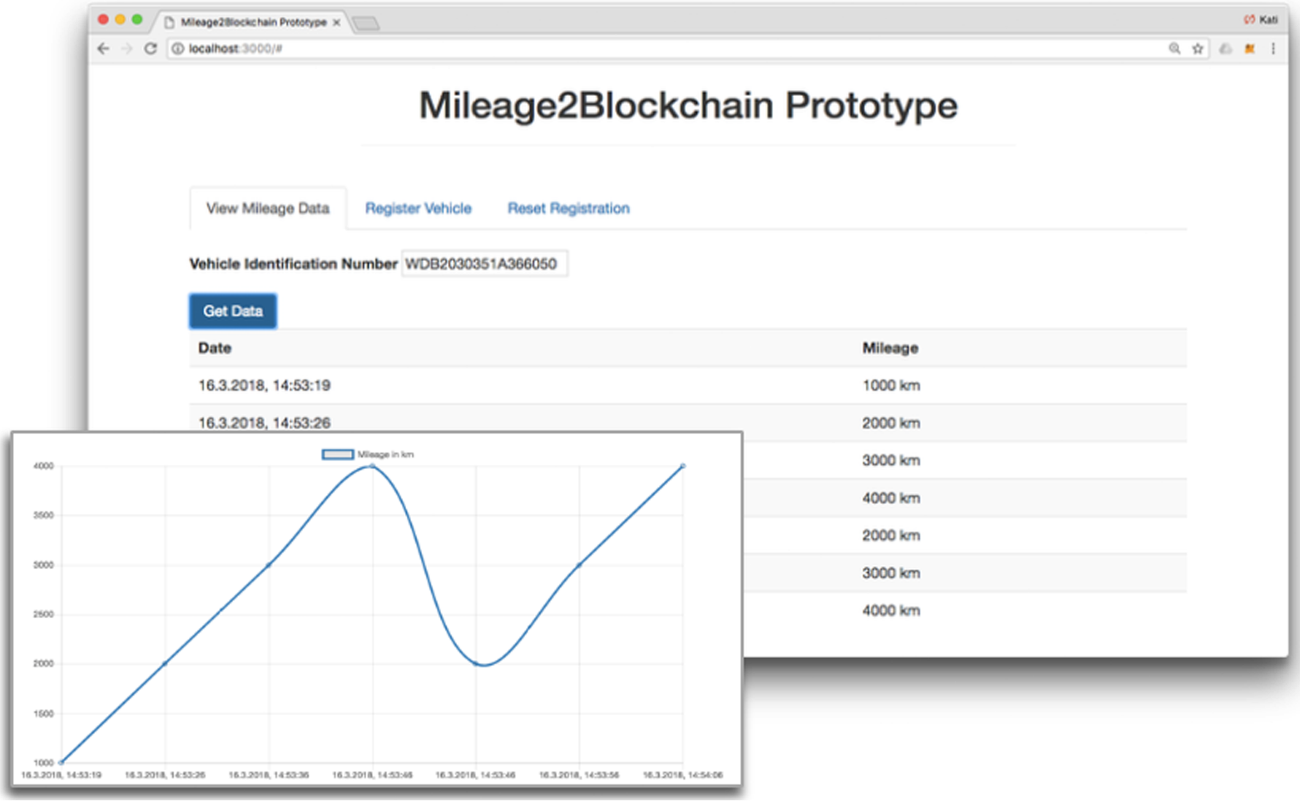
the miners in the Ethereum network, the transaction is confirmed. In the prototype setup, the block is of course mined directly by the only local Ganache-CLI node. A necessary prerequisite though is that the user charges the used account in MetaMask with Ethereum's cryptocurrency Ether.

The transaction is then visible within the blockchain that runs in a terminal window with information on the transaction hash, the gas usage, the block number, and block time (see Figure 7). In addition, the transaction history of the account can be seen within the MetaMask plugin.

If a user registers a false VIN, the operating party (ie, the smart contract owner) can reset the mapping by means of the web application. The transaction looks similar as the registration transactions described above. While the mileage data on the blockchain remains immutable, only the mapping between public key and VIN is overwritten.

Moreover, the web application is also used to retrieve the mileage history of a vehicle. The user needs to enter the VIN of interest. The blockchain is then queried with the VIN in order to relate to the corresponding public key and display the corresponding mileage data in the web application. It is shown in a table with the storage date and additionally illustrated with a graph; thus, manipulation can be easily revealed (compare Figure 8). For requesting the mileage history, no transaction on the blockchain is made, but the user still has to use a blockchain client such as MetaMask in order to connect to the blockchain.

In the next section, we will discuss the coverage of the initially defined requirements for the prototype.



Mileage2Blockchain Prototype

Vehicle Identification Number WDB2030351A366050

| Date | Mileage |
|---|---|
| 16.3.2018, 14:53:19 | 1000 km |
| 16.3.2018, 14:53:26 | 2000 km |
|  | 3000 km |
|  | 4000 km |
|  | 2000 km |
|  | 3000 km |
|  | 4000 km |

**FIGURE 8** The registration transaction is listed in the blockchain. After the transaction is confirmed by the only node of the local test chain, a block is created and viewable within the blockchain in the terminal window

## 5 | EVALUATION AND DISCUSSION

The evaluation of the prototype is based on the initially defined requirements from Section 3. In the following, these requirements are discussed along the prototype.

### 5.1 | Requirement 1: No restrictions in reading mileage histories (free of charge)

According to the first requirement, potential buyers of used cars should be able to read out the mileage history of a vehicle without barriers or significant costs.

Hence, the blockchain-based mileage storing system should be based on a public-permissionless blockchain. In contrast to a private blockchain, it allows all users to write and read transactions. No additional authorization system based on an intermediary or trusted third party is necessary, and users remain entirely anonymous.[42] Reading of transactions is possible for everyone free of charge in order to decrease barrier as long as they know the VIN.

However, we simplified the network in our prototype scenario and used a private network development environment in which only authorized nodes can read and write transactions in order to avoid transaction costs and energy consumption.[43] In addition, we successfully tested our prototype on a public Ethereum test net called Ropsten, where reading and writing is not restricted but open for the public. Ropsten, in general, acts very similar to the public Ethereum network. It also uses the PoW consensus mechanism; nonetheless, the used cryptocurrency Ether is worthless.[39,44] Consequently, our prototype could be replicated to the public-permissionless Ethereum blockchain that would be necessary for a scaled application. Hence, we regard this requirement as fulfilled.

### 5.2 | Requirement 2: No restriction in writing mileage histories (low transaction fees for car owners possible)

Following the previous requirement, writing mileage data on the blockchain should not be restricted in order to theoretically cover the entire used car market. Moreover, transaction costs should be low so that car vendors have an incentive to register their cars' mileage for signaling purposes and realize and increase their selling price.

As already described, the prototype was simplified by using a private network development environment called Ganache-CLI, where an infinite amount of Ether was provided for each account. Nevertheless, a shift to a real word scenario with a public-permissionless blockchain would allow everyone to write mileage data on the Blockchain. This leads definitely to transaction costs that are calculated by multiplying the gas usage (= total amount of consumed computational power, measured in gas) with the gas price (= price for one unit of gas, determined by the sender in order to incentivize miners). MetaMask suggests a standard gas price of 22 so called ‘‘GigaWei’’ (GWEI) as standard gas price. One GigaWei (GWEI) is correspondent to 1x10$^{-9}$ Ether. Storing a mileage data point would cost around 68 411 gas and thus leads to transaction costs of 0.001505 Ether, which would be €0.31 as of September 30, 2018 (however, Ether is highly volatile; hence, transactions costs can change very quickly). With a smaller gas price, eg, 8.5 GWEI, one transaction would cost only €0.12.[45]

In summary, these transaction prices are very low for car vendors in comparison to a price premium they can demand if they are signaling an immutable mileage data history. This economic sustainability of the prototype increases the incentive to register the vehicle on the blockchain application.

### 5.3 | Requirement 3: High usability of the user interface

We defined a user interface that allows to read mileage history and car registration as intuitive as possible so that many users without knowledge of the blockchain technology would be able to handle the mileage storing.

As explained in Section 2, blockchain as back-end can be extended to a decentralized application by adding a web application as front-end. Web clients that support Javascript APIs are used; thus, an interaction of the web browser to the blockchain was realized.[20] With the Mileage2Blockchain web application, the user can access the mileage data from any browser, that includes a web3 provider on any device. In addition to the web application where the data is entered, the registration of a car requires the usage of the browser wallet with balance in Ether. We used the browser extension MetaMask, which offers an intuitive user interface similar to traditional online payment system.[41]

Therefore, the prototype is considered to be handled by a large number of users without any deep knowledge of the blockchain technology. A usability test and survey could be used to further improve the usability.

### 5.4 | Requirement 4: Tamper proof track from data creation to the storage on the blockchain

The reliability of data entries is one of the major challenges of blockchain-based solutions for a system of records.[32,46] If data can be manipulated before its entry to the blockchain, a solution based on blockchain is not resistant to manipulation. In the scope of this paper, the prototype was simplified due to efficiency and financial reasons. Therefore, the automotive black box solution including a blockchain client was realized through several hardware components that can be a gateway for manipulation. Furthermore, the prototype has no explicit hedges (eg, scripts on the Raspberry Pi) that secure the immutability of data on its way to the blockchain. For example, the mileage could be manipulated every 999 km;

thus, the Mileage2Blockchain module would not send transactions anymore. Last but not least, data that comes into the blockchain via web application can be incorrect as well, eg, a vehicle could be registered with the public key of another one.

However, these problems can be addressed by a more complex realization of the mileage storing function based on our prototype that operates directly on an ECU. As explained before, data directly from an ECU is less open to attacks, similar to a black box. At this point, a blockchain integration guarantees the immutability of the data storage.[47] Consequently, under the assumption of a correct data entry based on an immutable ECU, the immutability characteristics of blockchain fulfil this requirement.

## 5.5 | Requirement 5: Completely automated process of mileage and timestamp storing

For the prototype, we strived for a completely automated technological solution without human interactions to reduce gateways for human errors, manipulation, and fraud.

Therefore, the business logic for data storing, the vehicle registration, and its resetting is realized in a smart contract. With its self-executing, deterministic, and automatically enforceable character, it eliminates the need of trusted third parties and human interactions. Consequently, it provides higher tamper protection and contractual security comparing to traditional database solutions.[19,25]

As explained before, the Ethereum client can directly run on a vehicle's ECU, which we simplified in the prototype with a simulated black box. Time-stamped mileage data is then automatically tracked and stored on the blockchain via the same smart contract functions described above without any human interaction in line with the requirement. The necessary prerequisite for a vehicle though is to have internet access. According to estimations by the VDA,[48] an estimated amount of 80% of new cars will be equipped with internet access. On the one hand, used cars often do not yet have internet connectivity, that provides a burden for a scaled application of the prototype. On the other hand, used cars can be retrofitted with a WIFI antenna which can be located in a black box. In case of internet access interruption, the blockchain client still works offline and sends transactions whenever the connection is rebuilt.

Another limitation is that the blockchain client can only be implemented directly on ECUs in new constructed cars because the blockchain client has to be integrated into the car's communication matrix according to industry experts. However, as mentioned above, used cars can be retrofitted with a black box similar as in the prototype.

## 5.6 | Requirement 6: Cost-efficient frequency and secure mileage and timestamp storing

Another requirement of the prototype is to store the mileage history, including a timestamp, securely and in a regular frequency while keeping transaction volume at a low level.

Firstly, the consensus mechanism of blockchain technology, in general, enforces that the integrity of the system is validated and verified as well as cryptography ensures the immutability and security of the data.[47,49] Secondly, in the prototype setup, the mileage and timestamp are stored every 1000 km according to the defined smart contract logic on the blockchain under the assumption that a car's value significantly decreases within these intervals. Hence, the requirement of secure and regular transmission of mileage data is fulfilled in the prototype scenario.

## 5.7 | Requirement 7: Verification of the mileage history with a VIN

The mileage history has to be assigned to the corresponding car, eg, by means of the VIN, so that interested parties that know the VIN can easily look up the mileage data before they purchase a vehicle. Moreover, tracking and sending mileage data to the blockchain should begin directly after the system is initially implemented on the vehicles' ECU.

As mentioned earlier, the mileage history, including timestamps, is stored on a private Ethereum development environment to keep things simple. However, the application is intended for a public-permissionless Ethereum blockchain where every network participant has access to the data due to the public ledger. Thus, the data is stored completely transparent and is additionally highly traceable.[14,45] Hence, it is possible to retrieve mileage data simply by entering the VIN in the user interface. However, it needs to be possible to delete personal data such as the VIN. Therefore, the hash of the VIN is stored separately from the mileage and timestamp data, while it is mapped to the vehicle's public key. It is ensured that mapping can be overwritten while mileage and timestamp data remain immutable on the blockchain. In this case, no mileage data would be displayed when entering the VIN in the front-end.

The web application guarantees that the user can check anywhere and anytime the mileage data of a vehicle registered on the blockchain if provided with the VIN.

## 5.8 | Requirement 8: Compliance to GDPR and anonymity of users

A blockchain-based mileage storing system needs to be compliant with data protection law such as the GDPR. In short, the owner of a car should be sovereign of their data and provide consent for mileage history tracking.

The mileage and timestamp data linked to a VIN is considered as personal data.[50] The GDPR requires a "*right to be forgotten*", hence users should be able to erase all personal data from a server or a network, but this conflicts with the immutability of data in a blockchain. As mentioned previously, only the hash value of the VIN is stored on the blockchain and the operating party (eg, the OEM) has the possibility to overwrite the mapping of the hashed VIN to the public key. Consequently, the VIN is not replicable and the mileage history not accessible via the web application. If the mileage data is not traceable using a VIN, it is not considered as personal data anymore and the "right to be forgotten" is guaranteed. Furthermore, the combination of a VIN and a public key is initially only known by the car owners themselves, consequently only the car owners can register their cars. Finally yet importantly, we required absolute anonymity of the users. First of all, the choice of implementing the mileage storing function on a public-permissionless blockchain provides the advantages that users of the Mileage2Blockchain web application are not under control of any CA.[42]

Consequently, users remain anonymous for mileage reading queries. Again, a blockchain-based solution is superior to a traditional central database, where a CA has to verify the authenticity of every user. However, against the initial paradigm of public permissionless blockchains, tools have been recently developed to facilitate a deanonymzation of blockchain addresses in order to identify its owners.[51] Hence according to current research, the users who send transactions (i.e. the car owners) remain only pseudo-anonymous. In conclusion, we consider our prototype compliant to the GDPR, however ongoing research and legal assessment based on court cases is necessary in order to assess the compliance to data protection regulation.

## 6 | CHALLENGES AND LIMITATIONS

In summary, all initially defined requirements were fulfilled by the prototype. Nevertheless, some design decisions reveal challenges and limitations. In particular, a public-permissionless blockchain, which is considered to be the most secure type of blockchains due to its totally distributed network and work-intensive consensus mechanisms, led to the following challenges and limitations that can be mostly considered as general problems of the Ethereum ecosystem.

### 6.1 | Limited throughput of the Ethereum network

The decision of using the public-permissionless Ethereum blockchain causes the problem of low throughput that impacts in turn the scalability and latency negatively.

The public Ethereum network currently has on average an expected throughput of around ten transactions per second due to the PoW consensus mechanism that requires high computing power.[35] In comparison, VISA has a throughput of 2000 to 10 000 transactions per second.[52] However, this is not a problem of the prototype per se but of the Ethereum blockchain, the prototype which it is based on. The planned introduction of a PoS-based consensus mechanism in the public Ethereum network promises a higher throughput than PoW and can reduce these limitations in the future.[33]

### 6.2 | Limited scalability of the application

Another severe problem of public-permissionless blockchains is the scalability that it is restricted by the throughput. It is unclear whether the large volume of mileage data of the entire used car market can be handled by the Ethereum-based prototype.

If an average transaction throughput of ten transactions per second is assumed for the public Ethereum network, the average annual amount of transactions adds up to 315.36 million transactions per year. Under the assumption that an average passenger car in Germany annually drives an estimated amount of 14 015 km,[53] one vehicle would send 14 transactions per year. Further assuming that all vehicles in Germany (45.80 million in 2017[54]) send their mileage data regularly to a blockchain, the network had to process approximately 641 million transactions. Based on these estimations, the public Ethereum network would only cover approximately 50% of the used car market just in Germany if no other Ethereum transactions would be processed.

The limited transaction throughput is a burden for the entire Ethereum ecosystem. Further improvements of the Ethereum performance (such as the aforementioned change of consensus mechanism) are expected though as this is a general problem of Ethereum.

### 6.3 | Unpredictable transaction costs

Unpredictable transaction costs are a further problem, as they depend on the gas price and the price for Ether that are highly volatile. There is no equilibrium gas price. The gas price to incentivize the miners to mine transactions is varying and cannot be forecasted. A higher gas price is expected to be necessary to incentivize more miners in order to validate the increasing amount of transactions in the Ethereum network. This would, of course, lead to higher transaction costs within the prototype. In addition, transaction costs also currently depend on the exchange rate of fiat currencies to the cryptocurrency Ether that is used for the payment of transactions.

### 6.4 | Limited and dependent latency

Limited and dependent latency is a major challenge of the Ethereum network and the blockchain technology.[55] Consequently, the arising limitations are reflected in our prototype as well. As described above, a higher gas price incentivizes miners to accelerate the processing of transactions; thus transactions, become confirmed faster. Moreover, it is limited by the network's throughput rate because the consensus mechanism needs a specific amount of time. In the presented use case, a delay of several seconds or minutes before confirming mileage data to the blockchain would not cause any problem because writing data in real-time is not required. In summary, the planned introduction of a Proof-of-Stake consensus mechanism in the public permissionless Ethereum blockchain will address these major limitations,[33] but still have to be proven in real-world scenarios. Therefore, we also consider other technological blockchain-basedsolutions: Different distributed ledger technologies such as the IOTA tangle, a directed cyclic graph for transactions storing, enable fast and scalable machine-to-machine transactions without the need for transactions costs.[56] Central database solutions do not meet the requirements of our prototype (e.g. R4), although they might perform better with respect to velocity.[31] Moreover, applications based on a private cloud might be less costly in executing business processes, particularly in comparison to public-permissionless blockchains such as Ethereum.[57] Nevertheless, all alternatives to our Ethereum-based prototype rely on a secure internet connection and a secure data transfer protocol as well. Additionally solutions such as IOTA's Tangles are less distributed (and can be considered as 'less trustful') as they depend on a coordinating node.[58] An extensive analysis of alternative technological approaches goes beyond the scope of the paper and is subject to future research such as in[31,57] and.[58]

## 7 | CONCLUSION

The aim of this paper was the construction of a trusted system of records that is based on blockchain technology to address the problem of asymmetric information on the used car market as described by Nobel laureate Akerlof[9] in his influential essay “The Market for Lemons.” Blockchain technology with its immutable and transparent ledger has the potential to address institutional problems such as information asymmetries. In particular, the mileage of used cars is a significant price and quality determinant and therefore often subject to undetected manipulation. Therefore, the storage of the mileage history of a used car should be immutable, transparent, and accessible to a potential buyer in order to balance the asymmetric information.

We defined eight requirements for a technological solution based on blockchain technology and developed a prototype to demonstrate the feasibility of a secure distributed mileage storing system. The inner functional logic and design of the prototype can be reproduced to the development of any other blockchain-based system of records in other application areas that are characterized by lack of trust between actors or by the absence of a trusted central authority (eg, land registries).

The prototype stores the mileage and timestamp in a secure and cost-efficient frequented way on a distributed database based on blockchain technology. The storing process is completely automatic, does not require any human interference and once set up, does not require any trusted third party due to the deployment on the public Ethereum network and the implementation of smart contracts. The mileage data is accessible and transparent to anyone who knows the vehicle identification number (VIN), if the VIN was registered on the blockchain by a vehicle owner. The data can be retrieved free of charge by entering the VIN in an intuitive web application so that users do not need knowledge of the underlying blockchain technology. Therefore, no central authority that needs to be trusted is required to validate the database. The mileage data on the blockchain remains immutable and is stored in a distributed database which makes it robust and secure against manipulation or potential hackers in comparison to a central database. However, in order to comply with data protection and guarantee anonymity of the users,the mapping of a VIN to a mileage data entry on the Blockchain can be overwritten.

The prototype is based on two simplifying assumptions that do not undermine a potential realization as a scaled application. First, we use a private development environment with only one mining node in order to avoid transaction and energy costs, whereas a scaled application would require a public-permissionless blockchain. Second, the prototype is implemented on a simulated black box and not directly on a vehicles' ECU to reduce development costs in the scope of this paper. In a future application, an implementation directly on an ECU of newly manufactured cars as an industry-wide standard is essential to guarantee a tamper proof and secure transmission of mileage data to the blockchain. Therefore, a diffusion on the used car market would either be lagged in time or used cars needed to be retrofitted with a certain black box similar to the one we have used in the prototype. In addition, an implementation of a blockchain-based mileage storage system requires the development of a business model or further regulatory measures because car manufacturers need incentives to implement a transparent and secure technological solution. Moreover, a scaled application is also depending on an operating party that issues the smart contracts and private/public keys. These incentives might be trigger by other markets than the used car market that would benefit from a secure and tamper proof mileage storing as well, eg, from leasing companies to control and adjust leasing rates, insurances.

Nevertheless, a series of challenges and limitations mostly of the Ethereum network and ecosystem itself needs to be solved for a potential scaled application. First, the throughput rate of the public-permissionless Ethereum blockchain is limited due to the PoW consensus mechanism that requires considerable computing power. This results in a high latency and limited scalability. According to our estimations, the network could not even cover the German used car market under the assumption that all vehicles in the country send a transaction every 1000 km (according to estimations, the average annual mileage is 14 000 km). In addition, the latency depends also on the transaction costs that are determined by the gas price. The higher the gas price, the more incentivized the miners are to process the transaction. It remains a challenge to define the ideal gas price to ensure a feasible validation time that will also fluctuate. The gas price is further volatile as it is expected to increase if the Ethereum network is increasing. Moreover, the price of gas is measured in the cryptocurrency Ether, whose exchange rate to fiat currencies is highly volatile as well.

The planned transformation of Ethereum from PoW to PoS consensus mechanism is expected to address these limitations. It requires less computational power in a future scenario because the likelihood that a peer creates a block depends on the amount of coins (ie, stakes) a miner owns and not on computational power.[19] Further distributed ledger technologies such as the IOTA protocol have to be explored for a potential application. In particular, permissioned blockchains might overcome the throughput as well as latency challenges and avoid transaction costs. However, a private-permissioned blockchain platform such as Hyperledger Fabric limits the participation in the network to only known and trusted nodes and requires an additional system respectively trusted central authority for the authorization of users. This would contradict one of our requirements that every potential buyer on the used car market should be able to read out the mileage history of a vehicle without barriers or significant costs.

The prototype has demonstrated the potential of blockchain technology to bridge the trust gap by building a trusted system of records, which might be particularly relevant for emerging countries (e.g. land registries). Further research can replicate the prototype or the immanent logic to other use cases that require a tamper proof and trusted system of records, explore business models and the extraction of specifically with respect to the automotive use cases harness further vehicle data from ECUs to a blockchain-based database.

## ORCID

*Moritz Böhmecke-Schwafert* https://orcid.org/0000-0003-0543-2622

## REFERENCES

1. Yan Z, Zhang P, Vasilakos AV. A survey on trust management for Internet of Things. *J Netw Comput Appl*. 2014;42:120-134.
2. Samaniego M, Deters R. Blockchain as a service for IoT. Paper presented at: 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData); 2016; Chengdu, China.
3. Huh S, Cho S, Kim S. Managing IoT devices using blockchain platform. Paper presented at: 2017 19th International Conference on Advanced Communication Technology (ICACT); 2017; Bongpyeong, South Korea.
4. OECD Science, Technology and Innovation Outlook 2016. Paris, France: OECD; 2016.
5. Christidis K, Devetsikiotis M. Blockchains and smart contracts for the Internet of Things. *IEEE Access*. 2016;4:2292-2303.
6. Macdonald M, Liu-Thorrold L, Julien R. The blockchain: a comparison of platforms and their uses beyond bitcoin. 2017.
7. Bamberger B. *Blockchain-Technologie: Vom Hype zur Wirklichkeit*. Frankfurt, Germany: Frankfurt School Blockchain Center; 2017:1-14.
8. Drescher D. *Blockchain Basics: A Non-Technical Introduction in 25 Steps*. Frankfurt, Germany: Apress; 2017.
9. Akerlof GA. The market for "lemons": quality uncertainty and the market mechanism. *Q J Econ*. 1970;84(3):488-500.
10. ADAC. ADAC Initiative gegen Tacho-Betrug. 2017. https://www.adac.de/infotestrat/fahrzeugkauf-und-verkauf/gebrauchtfahrzeuge/tacho-manipulation/default.aspx. Accessed October 02, 2019.
11. EU Regulation, 2017/1115-2.3. 2017.
12. Wurster S, Böhmecke-Schwafert M, Hofmann F, Blind K. Born global market dominators and implications for the blockchain avantgarde. In: *Corporate and Global Standardization Initiatives in Contemporary Society*. Hershey, PA: IGI Global; 2018.
13. CoinDesk. $6.3 Billion: 2018 ICO Funding Has Passed 2017's Total. 2018. https://www.coindesk.com/6-3-billion-2018-ico-funding-already-outpaced-2017/. Accessed October 02, 2019.
14. Wright A, De Filippi P. Decentralized blockchain technology and the rise of lex cryptographia. *SSRN Electron J*. 2015.
15. Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. 2008.
16. Ko V, Verity A. Blockchain for the humanitarian sector: future opportunities. Digital Humanitarian Network. 2016.
17. Voshmgir S. *Blockchains, Smart Contracts und das Dezentrale Web*. Berlin, Germany: Technologiestiftung Berlin; 2016.
18. Szabo N. Smart contracts: building blocks for digital markets. 1996.
19. Morabito V. *Business Innovation Through Blockchain: The B3 Perspective*. Cham, Switzerland: Springer International Publishing AG; 2017.
20. Buterin V. *A Next Generation Smart Contract & Decentralized Application Platform*. White Paper. Bern, Switzerland: Ethereum Foundation; 2014.
21. Bit Fury Group, Garzik J. *Public Versus Private Blockchains: Part 1: Permissioned Blockchains*. Version 1.0. White Paper. Amsterdam, The Netherlands: Bitfury Group Limited; 2015.
22. Gu Q, Liu P. Denial of service attacks. In: *Handbook of Computer Networks: Distributed Networks, Network Planning, Control, Management, and New Trends and Applications*. Vol 3. Hoboken, NJ: John Wiley & Sons; 2012.
23. Farell R. An analysis of the cryptocurrency industry. *Whart Res Sch Journal Pap*. 2015;130.
24. Peters GW, Panayi E. Understanding modern banking ledgers through blockchain technologies: future of transaction processing and smart contracts on the internet of money. CoRR. 2015. https://arxiv.org/abs/1511.05740
25. Mattila J. *The Blockchain Phenomenon - The Disruptive Potential of Distributed Consensus Architectures. Berkeley Roundtable on the International Economy*. Working Paper. Berkeley, CA: University of California, Berkeley; 2016.
26. Bentov I, Lee C, Mizrahi A, Rosenfeld M. Proof of activity: extending Bitcoin's proof of work via proof of stake. 2013;240258:1-19.
27. Brauckmann J. TÜV Rheinland Studie Tachomanipulation. 2015.
28. Fahrzeugdiagnose Informationsportal. OBD-2.net. 2017. https://www.obd-2.de/#. Accessed October 02, 2019.
29. Carly - Connected Car. Datenblatt Gebrauchtwagen-Check. 2016. http://www.mycarly.com/press-deutsch/datenblatt-zum-gebrauchtwagen-check/. Accessed October 02, 2019.
30. TachoSpion. Sicherheit beim Gebrauchtwagenkauf: Verschleißgrad auf einem Blick. 2017. http://www.tacho-spion.de. Accessed October 02, 2019.
31. Chowdhury MJM, Colman A, Kabir MA, Han J, Sarda P. Blockchain versus database: a critical analysis. Paper presented at: 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; New York, NY.
32. Lemieux VL. Trusting records: is blockchain technology the answer? *Rec Manag J*. 2016;26(2):110-139.
33. Buterin V, Griffith V. Casper the friendly finality gadget. 2017:1-10.

34. Luu L, Chu D-H, Olickel H, Saxena P, Hobor A. Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security; 2016; Vienna, Austria.
35. Buterin V. Ethereum platform review: opportunities and challenges for private and consortium blockchains. 2016.
36. Bashir I. *Mastering Blockchain: Deeper Insights Into Decentralization, Cryptography, Bitcoin, and Popular Blockchain Frameworks*. Birmingham, UK: Packt Publishing; 2017.
37. Consensys. Truffle - Documentation. 2017. http://truffleframework.com/docs/. Accessed October 02, 2019.
38. Consensys. Bundling with Webpack. 2017. http://truffleframework.com/tutorials/bundling-with-webpack. Accessed October 02, 2019.
39. Dannen C. *Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners*. New York, NY: Apress; 2017.
40. Svensson C. Web3j Doc - Solidity Smart Contract Wrappers. docs.web3j.io. 2017. https://docs.web3j.io/getting_started/. Accessed October 02, 2019.
41. MetaMask. MetaMask - Brings Ethereum to your browser. 2017. https://metamask.io. Accessed October 02, 2019.
42. Buterin V. On Public and Private Blockchains. Ethereum Blog. 2015. https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/. Accessed October 02, 2019.
43. Lewis A. A gentle introduction to blockchain technology. Auckland, New Zealand: Brave New Coin. 2015:1-13.
44. Etherscan. Ropsten Etherscan - The Ethereum Block Explorer. 2018. https://ropsten.etherscan.io. Accessed October 02, 2019.
45. CoinMarketGap. Ethereum (ETH) market capitalization. https://coinmarketcap.com/currencies/ethereum/. Accessed October 02, 2019.
46. Hofmann F, Wurster S, Ron E, Böhmecke-Schwafert M. The immutability concept of blockchains and benefits of early standardization. Paper presented at: 2017 ITU Kaleidoscope: Challenges for a Data-Driven Society (ITU K); 2017; Nanjing, China.
47. Gupta M. *Blockchain for Dummies*, IBM Limited Edition. Hoboken, NJ: John Wiley & Sons Inc; 2017.
48. VDA. Anteil der mit dem Internet vernetzten Neuwagen in den Jahren 2015 und 2017. statista.com. 2017. http://de.statista.com/statistik/daten/studie/407955/umfrage/anteil-der-mit-dem-internet-vernetzten-fahrzeuge/. Accessed October 02, 2019.
49. Seibold S, Samman G. Consensus: immutable agreement for the internet of value. Amstelveen, The Netherlands: KPMG; 2016.
50. Schulzki-Haddouti C. Autoindustrie und Datenschützer: KfZ-Daten unterliegen dem Datenschutz. Heise Online. 2016. http://www.heise.de/newsticker/meldung/Autoindustrie-und-Datenschuetzer-KfZ-Daten-unterliegen-dem-Datenschutz-3084253.html. Accessed October 02, 2019.
51. Klusman R, Dijkhuizen T. Deanonymisation in Ethereum Using Existing Methods for Bitcoin. 2018.
52. Swan M. Blockchain thinking: the brain as a DAC (decentralized autonomous corporation). *IEEE Technol Soc Mag*. 2015;34(4):41-52.
53. Kraftfahrt-Bundesamt. Gesamtkilometer steigen um 1,4 Prozent. 2016:1-2.
54. Kraftfahrt-Bundesamt. Zentrales Fahrzeugregister (ZFZR). 2014:2833.
55. Yli-Huumo J, Ko D, Choi S, Park S, Smolander K. Where is current research on blockchain technology?—a systematic review. *PLOS ONE*. 2016;11(10):1-27.
56. Popov S. The Tangle - IOTA whitepaper v1.4.3, 2018.
57. Rimba P, Tran AB, Weber I, Staples M, Ponomarev A, Xu X. Quantifying the cost of distrust: comparing blockchain and cloud services for business process execution. *Inf Syst Front*. 2018.
58. Salimitari M, Chatterjee M. A Survey on Consensus Protocols in Blockchain for IoT Networks.

**How to cite this article:** Preikschat K, Böhmecke-Schwafert M, Buchwald J-P, Stickel C. Trusted systems of records based on Blockchain technology - a prototype for mileage storing in the automotive industry. *Concurrency Computat Pract Exper*. 2020;e5630. https://doi.org/10.1002/cpe.5630

# APPENDIX

## A.1 | Tables

**TABLE A1** "Four versions of the blockchain as a result of combining reading and writing restrictions' (modified according to the work of Macdonald et al[6])

| | | Reading access and creation of transactions | |
|---|---|---|---|
| | | Everyone | Limited |
| Writing access of blocks | Everyone | Public and permissionless | Private and permissionless |
| | Limited | Public and permissioned | Private and permissioned |

**TABLE A2** Overview of the requirements for a blockchain-based system of records

| No | Requirement |
|---|---|
| R1 | No restrictions in reading mileage histories (free of charge) |
| R2 | No restriction in writing mileage histories (low transaction fees for car owners possible) |
| R3 | High usability of the user interface |
| R4 | Tamper proof track from data creation to the storage on the blockchain |
| R5 | Completely automated process of mileage and timestamp storing |
| R6 | Cost-efficient frequency and secure mileage and timestamp storing |
| R7 | Verification of the mileage history with a VIN |
| R8 | Compliance to GDPR and data protection |

**TABLE A3** Overview of smart contract functions, including its application, access, and description. The marked functions (*) do not need to be implemented explicitly

| Functionality | Smart contract function | Access | Description |
|---|---|---|---|
| **Data storing** | *storeMileage (mileage, timestamp)* | Writing everybody | Stores mileage and timestamp, mapped to the caller's public key |
| **Registration (mapping)** | *mapVinToPublicKey (hashedVin, publicKey)* | Writing everybody | Stores mapping of hashed VIN to public key. If a mapping to hashed VIN already exists, an error is returned |
| **Resetting** | *resetVinMapping* | Writing owner of smart contract | Resets an existing mapping of hashed VIN to public key |
| **Reading** | *getPublicKey (hashedVin)**<br>*getMileage (publicKey)** | Reading everybody | Returns mapping of hashed VIN to public key<br>Returns history of mileages mapped to public key |

## A.2 | Technical Specifications of FlexDevice L and M

**FlexDevice L:**

- ARM Cortex A9 dual core (800 MHz) with 1 GByte DDR3 RAM
- 10 bus channels (via pluggable transceivers)
- Up to 7 Ethernet channels
- Up to 10 CAN channels
- Up to 8 FlexRay channels (4 channel A and 4 channel B)
- WiFi and Bluetooth extension available
- Wake-up/sleep mode support
- Startup <200 ms possible
- IP67
- Temperature range - 40°C to +85°C

**FlexDevice M:**

- PowerPC MPC5567 (120 MHz) with 2 MB SRAM
- 5 bus channels (4 via pluggable transceivers)
- Up to 5 CAN channels
- Up to 4 FR channels
- Up to 2 LIN channels
- Wakeup/Sleep mode support
- Startup <100 ms
- Complete development environment included
- PC connection via Ethernet
- IP67
- Temperature range -40°C to +85°C