Rheinisch-Westfälische Technische Hochschule Aachen
Informatik 5, Information Systems
Prof. Dr. Stefan Decker

Master/Bachelor Thesis Proposal

Privacy-Preserving Storage of Vehicle Data on the Blockchain

Lukas Malcher

August 23, 2018

Advisor Benjamin Heitmann, Ph.D.
Supervisor Prof. Dr. Stefan Decker

Abstract

Today's cars offer rich sensor platforms allowing to monitor the vehicle's current values and overall health. Telematics describe the method of capturing, uploading and processing such data. The data is usually stored in the cloud of the manufacturer or black boxes are installed for this purpose. These services are bound to specific parties, are not transparent and often do not respect the privacy of the user. This proposal introduces future work of a thesis that aims to eliminate these issues by designing an open system backed by the blockchain technology that supports the privacy preserving collection of automotive telematics. The blockchain offers an immutable ledger with a strict consensus and the usage of ring signatures keep the users anonymous. Possible use cases with their requirements and potential solutions are prepared and various methods of benchmarking the system to reveal any bottlenecks are proposed. Finally, an implementation draft of a simulation is introduced which should test the feasibility of such a system.

Contents

1 Introduction

1.1 Motivation

In the recent years the *Sensor Web*, basically sensors that are accessible from the Internet, evolved to the Internet of Things (IoT) where sensors themselves are connected to the Internet and communicate independently. Vehicles are no exception and can be considered as IoT devices as well. Nowadays they provide rich sensor platforms that regularly upload data such as GPS locations, sensor data or road conditions to the cloud. The vast amount of data allows various applications ranged from comfort to efficiency and safety services [Ger+14].

Unfortunately the privacy of the user is often not respected and location based data which can be linked to personal data get collected [Jes13]. Instead of uploading the data to a cloud owned by specific companies, a new technology, the blockchain, could potentially eliminate privacy issues and also speed up processes between multiple parties in a cheap way. A blockchain deployment that is secure, scalable and open would constitute an international database with data in strict consensus. Then, for instance, a complete life-cycle of a vehicle could be recorded, from its manufacturing, all its accidents and repairs to its final demolition. The level of transparency and immutability would allow buyers to quickly and reliable calculate resale prices. However, while some applications might require to identify the vehicles, in general, user tracking should be avoided.

1.2 Thesis Goal

This thesis studies the feasibility of using the blockchain technology to store and process sensor data collected by vehicles. The blockchain will be used as an append-only privacy preserving database containing a global pool of knowledge which is open for the public and can be used by non-trusting third parties in a fast, transparent and secure way. The resulting system provides several use cases including validated storage for a vehicle's life-cycle (containing maintenance data, age, driving performance, etc.), storage of driving performances enabling flexible insurances, gathering traffic statistics and notifications as well as general learning purposes enabling a global swarm intelligence. Compared to traditional solutions, the use of a blockchain based system potentially eliminates various issues such as general privacy concerns, slow processing times between multiple parties and trustless validation. Several approaches to emerging problems will be examined with the focus on benchmarking limitations in regards to scalability, user privacy and information validity. But in order to succeed and to be superior to traditional solutions the resulting architecture and protocols must meet multiple requirements.

The primary goal is to determine whether it is feasible to use a blockchain based solution for automotive use cases or what factors hold back the blockchain technology to become a new alternative for distributed database applications. A proof of concept implementation should demonstrate the possibilities as well as the drawbacks and emphasize the final conclusion.

1.3 Outline

The following sections will start with the general concepts and goals of distributed databases. This will lead to the blockchain approach with its specific properties that are discussed. Related to the use case, possible privacy and information validity attacks with their solutions are examined. Then, a blockchain specific approach for automotive use cases is discussed in Section 2.4. The main use case with its requirements is defined and a fitting blockchain technology is chosen in Section 3. Finally, Section 4 and 5 present a conceptual approach and the plan of implementation. Section 6 finishes with the general plan of the project.

2 Background and Related Work

In order to provide a deeper insight of the possible opportunities and challenges, the following section will begin with a discussion about traditional distributed database systems. Thereafter, the specific properties will be compared to the blockchain technology and valued accordingly.

2.1 Techniques of Distributed Databases

Companies with big applications might want to scale their database system to keep up with the vast amount of traffic data that need to be stored. Scaling horizontally, by deploying a cluster, provides several advantages over a single instance. Firstly, a cluster can handle more throughput than a single instance would be capable of. Secondly, high availability can be ensured so that another instance can take over a failed one. And lastly, latency can be minimized by distributing instances to several regions (localization) [Mul].

In general, distributed database systems are heavily modified and optimized for specific use cases. There is no common way how to provide storage for Big Data applications. The definition[1] of a distributed database boils down to

1. a database that is not entirely stored at a single physical location, but rather is dispersed over a network of interconnected computers.

2. a database that is under the control of a central database management system in which storage devices are not all attached to a common processor.

The following basic techniques are available that distributes the data (based on examples in relational databases).

Fragmentation, also often referred to sharding, breaks up the data into logical units. In a relational database, such a logical unit could be for example a table describing a relation between multiple entities. More specifically, *horizontal fragmentation* divides the relation by grouping rows in a logical manner. As an example, let a relation declare which employee works at what department. Then, a fragmentation could group the data of the tables `EMPLOYEE` and `WORKS_AT` by department. As a result, the corresponding fragments would be stored in a local database within the building where the department is located. Since the data is location related it is likely that most queries will affect mostly local data, increasing the performance of the queries.

Vertical fragmentation divides a relation by columns. The `EMPLOYEE` table would be split up by e.g. personal information and work related data. Depending on the underlying data and use case, the fragmentation must be defined individually and can be a mix of horizontal and vertical fragments.

Another technique is the process of *replication* which synchronizes the data to multiple instances. This improves availability and reliability remarkably if it is done sufficient because then, instances can fail without having downtimes or data loss. Performance is improved as well, especially in regards of read operations because row retrieval queries can be executed locally. On the downside, update queries must be applied to all replicas which slows down the process significantly [EN10].

Compared to a simple one-machine deployment a more complex management system is required. When it comes to write operations the system needs to decide where those operations are performed and how the changes get synchronized to all other replications. A rather simple solution, the *master-slave replication* defines a single database instance as a master node. Changes on the data are applied exclusively on the master node and then

[1] `https://www.its.bldrdoc.gov/fs-1037/dir-012/_1750.htm`

synchronized to the slaves. While this is an efficient and simple way to scale read-heavy applications, unfortunately it neglects the original intend of the replication. The master node might become a bottleneck regarding throughput, reliability and latency [Mul].

Compared with that, in a *multi-master replication*, all instances are considered as authoritative, hence, each database instance is a master node and can perform write operations. Synchronization is done peer-to-peer and requires a more complex concurrency control technique. These techniques avoid conflicts when multiple transactions work on the same data simultaneously. For example, the use of locks ensures that only one transaction can update specific data at the time. Instead of locking the whole database, typically only as little data (or rows) are locked for as short time as possible. A conflicting transaction will need to wait until the lock is released [Mul].

Another technique is called Multiversion Concurrency Control (MVCC). At each point in time a snapshot of the data is considered to be in consensus. New transactions will only see the current snapshot even if another transaction is altering the data at the same time. Ultimately, MVCC tries to detect and prevent conflicting transactions which could mean that one of the conflicting transactions gets denied when a new snapshot is generated [Mul].

However, in a multi-master replication system this issue is especially critical because data can be altered in different instances simultaneously. A very elegant solution is the blockchain technology that provides a distributed MVCC [Mul].

2.2 Using the Blockchain as a Shared Ledger

In general, most applications utilize a database where the corresponding dataset is owned and managed by a single party. However, an issue arises when multiple parties want to collaborate and work on a single dataset.

Such a case would be, for instance, a dataset that is shared by multiple license/certification offices located all over the world. Offices could speed up inter-office processes if they are able to check, validate and update data that was issued by other parties in a fast and reliable way.

Instead of having multiple isolated databases, each providing a different interface, a distributed database system could be applied where each office would maintain a database node. The used data schemes would be standardized by design and replication as well as localized fragmentation can be applied accordingly. Unfortunately there is a critical drawback that need to be addressed.

Since everyone has write permissions, one party could decide to alter or delete data. Even if an append-only database system is used, also commonly referred as a ledger, a global consensus method is required that allows all participants to verify and accept the integrity and immutability of all data. Full transparency would help to avoid corruption and censorship.

Considering that multiple independent, potentially non-trusting, offices from several countries are involved, one possible solution would be a customized consortium blockchain database system which can be seen as a distributed ledger in form of an append-only multi-master fully replicated database that is in no need for a central administrator.

Gideon Greenspan, the CEO of the company behind the MultiChain platform, proposed several conditions that should be met by blockchain use cases [Blo; Mul].

2.2.1 The Database

Traditional solutions usually make use of relational or NoSQL databases consisting of a structured repository of information. Changes issued by transactions are done directly within the rows of the corresponding tables. This method is perfectly fine for most applications which are controlled by a single company. However, as stated in Section 1.1, the collected

data should be shared by multiple untrusted parties. As a possible solution, the blockchain is an open distributed database that could be used transparently and trustless by multiple untrusted or even competing parties. Historical data for statistical analyses are preserved by design.

2.2.2 Multiple Writers

In centralized solutions users have to register and authenticate themselves in order to get write permissions. This causes privacy concerns when location based information is involved. Issues like individual user tracking are included as [Wan+16] successfully demonstrated. Using the blockchain allow clients to stay anonymous and data consumers do not necessary need to know their clients.

2.2.3 Absence of Trust and Disintermediation

Since no single party have the control over incoming transactions it is not possible to keep specific data secret or sensor it. Participants cannot take advantages or manipulate datasets issued by others. They share the maintenance costs and benefit from it equally. This results in lower costs, more transparency, faster processes and generally, a bigger user base and thus, better services for all users.

2.2.4 Transaction Interaction

Transactions on the blockchain usually depend on other transactions issued by other participants. In Bitcoin for example this constitutes the most revolutionary component. In order to calculate ones assets, all transactions with unspent outputs belonging the corresponding address must be aggregated. There is no such table entry revealing the current balance of an address. The use case in question indicates similar characteristics. That a vehicle broadcasts an accident might be wrong information. How can one validate information of an untrusted party without the ability to check whether it is true or false? A possible idea would be a voting system where nearby participants create transactions referring to the initial one agreeing or disagreeing with the subject. A trust-factor would allow others to interpret the truth.

2.2.5 Set the Rules

A huge amount of unknown writers are authorized to broadcast transactions. Possible attacks like spamming transactions which cause false information or decrease the system performance should not be applicable. This is especially difficult to solve when no fees are involved. Not a condition per se but as a consequence of the previous points, Greenspan rises the need of regulations in form of rules. In order to get validated and inserted into the blockchain a transaction must be checked against a specific rule-set that all nodes have to follow. Those rules could handle the validation of real and fake vehicles, location proofs and double voting.

2.2.6 Selection of Validators

Nodes in a blockchain architecture have less power compared to a single central instance which prevent them of modifying or censoring transactions. In a fully decentralized blockchain everyone can maintain a validator node and nobody has more power than the others (besides 51% attack scenarios). However, in a consortium blockchain, validators must be chosen carefully because unsuitable compositions could allow biased nodes to favor conflicting

transactions or dismiss them completely. Other than Proof-Of-Work (PoW) or Proof-Of-Stake (PoS) based systems, manipulation is cheap and immutability is only achieved by mistrust between the validators. A well selected set of validator nodes complying with the requirements is discussed in Section 2.4. It basically consists of vehicle license offices located all over the world. Thus, the blockchain will be maintained in a group acting fair by design. This also allows the blockchain to survive situations such as sudden discontinued offices.

2.3 Information Validity Attacks and Privacy Concerns

The collection of automotive telematics usually includes location based data. Related to that, back in 2013, the paper [Jes13] explained the methods of generating traffic statistics by frequently collecting Global Positioning System (GPS) locations and speed values of users through smartphone apps. With sufficient data, mapping services are able to interpret various information such as traffic jams and points of interests. However, the quality of the interpretation depends heavily on the validity of the data which the user provides.

Unfortunately the GPS protocol is rather slow, does not guarantee to work in closed buildings/tunnels, is unencrypted, has no proof of origin or authentication features. These vulnerabilities led to various attacks [Cor18]. The main issue is based on the fact that GPS is a passive protocol which means that the user is able to determine his own positions but the satellites are not able to sign and approve gathered locations [Len+08]. There is no way one can proof that the claimed GPS coordinates from an untrusted party is in fact true or not.

This is why Google does not only focus on GPS coordinates as a single source of localization but combines it with trusted location positions. When Street View was created Google did not only take images from streets but also collected information about positions and MAC addresses of wireless access points. This resulted in data which combines trusted coordinates with positions of public available wireless routers that can be seen by all smartphones nearby. Hence, even though GPS settings are turned off, theoretically, Google is still able to calculate the exact position of the device. This method works reliable as intended, especially in bigger cities and it enhanced mapping quality in general. However, it does not prevent attackers from performing replay attacks. Once the attacker had driven the target route while collecting data, he was then able to replay the tour multiple times with virtual fake vehicles causing real traffic jams [Wan+16]. The described attacks affected multiple services such as Google and Waze. It is not clear if the issues are still up to date but the blockchain approach will definitely face them as well.

To avoid these kinds of attacks, previous works such as [Wan+16] introduced an online algorithm based on a *proximity graph* which forms a network of trust over time. The idea is that virtually generated fake vehicles are unlikely to come into proximity with real physical participants. Vehicles nearby to each other form *collocation edges* that represent an evidence of real physical objects. Fake vehicles controlled by the attacker can only form collocation edges with other fake vehicles or with the attacker's real vehicles. This results in a graph with only a few connections between real and fake nodes. Figure 1 shows such a proximity graph with some real vehicles and some malicious ones colored in red.

This approach requires a set of trusted nodes in order to bootstrap initial trust in the network since no participant can be trusted when the proximity graph is in its infant state. Such nodes could be WiFi access points or cellular stations where vehicles will have to come into physical proximity in order to get authenticated to create new collocation edges. All vehicles are fitted with WiFi devices leveraging the limited transmission rates of only a few meters. The WiFi device broadcasts a specific time-varying Service Set Identifier (SSID). This could be a unique string which cannot be guessed or manipulated by others generated by an algorithm similar to a Time-based One-Time Password algorithm (TOTP). With this method a vehicle A can proof its proximity to another vehicle B by sending B's unique

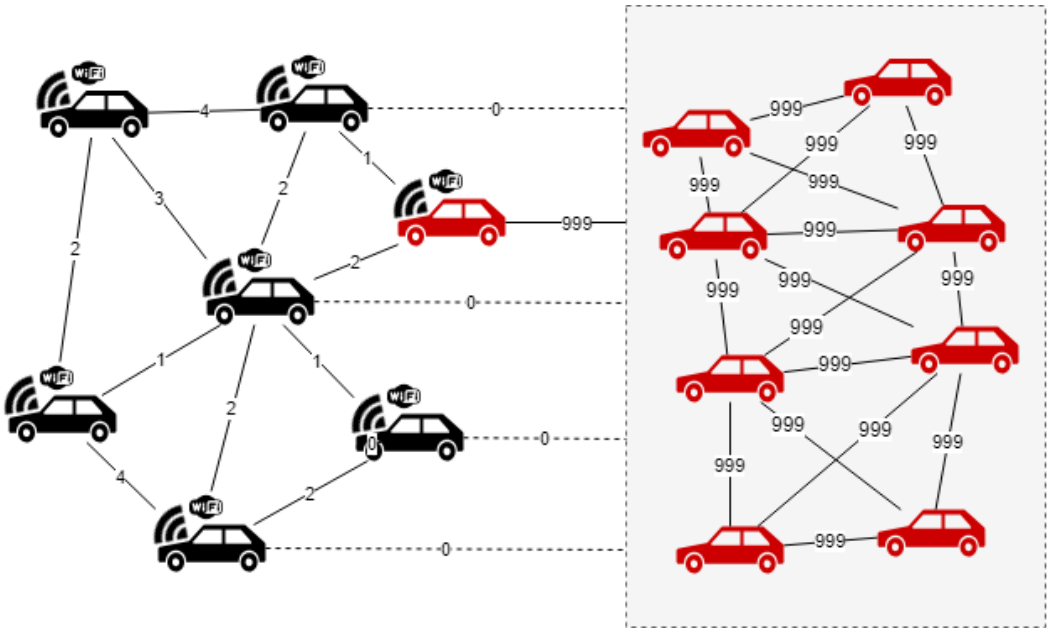

Figure 1: A proximity graph showing physical real vehicles and fake vehicles colored in red. The weight of the edges determine the amount of encounters in close ranges. An attacker tries to inject a virtual fake fleet by giving his network high weights. Note, that only the attacker's own physical vehicle is able to have weighted edges with his fake fleet. All other more trusted participants cannot have any encounters with them. Thus, sibyl subgraphs are detectable with community-detection algorithms quite easily [Wan+16]

time-specific string (which A can only know by being in close range to B) to the service provider who then can validate and update the graph [Wan+16]. This indicates that this approach is in need of a trusted central authority which makes it impractical for a blockchain usage without giving up privacy preserving methods.

Nonetheless, despite having no proof of location, mapping services seem to work very well. Even if an attacker tries to generate virtual traffic jams by spamming the mapping service with slow moving fake vehicles he cannot avoid that real vehicles send correct traffic data. This will cause conflicted information whether there is a traffic jam or not which is detectable easily. Anyway, attacks like these should be avoided by any means.

A more important aspect is the privacy of the user. In general, collecting personal data that allow tracking possibilities is considered as undesirable by the user [Jes13]. Unfortunately the attacks introduced in this section indicate that some form of authentication and authorization is required because blindly accepting data from unknown sources could result in serious traffic collapses.

Observations done by [Jes13] conclude that in case of Google Maps[2] and Waze[3], the anonymity of the user is not assured. Waze requires authentication and data packages sent to Google contain tokens that uniquely identify the user. The same applies to INRIX[4]; personal information are collected and stored for a long period of time [Inr]. TomTom[5] on the other hand requires user related data as well but promises to delete any personal

[2] `https://www.google.de/maps`
[3] `https://www.waze.com/`
[4] `http://inrix.com/products/traffic`
[5] `https://www.tomtom.com/en_us/drive/tomtom-traffic`

information shortly after [Tom]. As a conclusion, there is probably no such service that respects the privacy of the user at the time of writing this thesis.

2.4 Blockchain Architecture for IoT Devices and Smart Vehicles

IoT devices are usually devices with low computational power and capacity not fulfilling the requirements of a typical blockchain. The Lightweight Scalable Blockchain (LSB) is an architectural approach that allows IoT devices to benefit from blockchain properties such as end-to-end security and privacy without changing the blockchain technology itself [DKJ17]. Unfortunately there is no implementation of it currently. This concept can be extended for automotive use cases and works as follows [Dor+17].

Internet connected nodes of all kinds such as smart vehicles, smart homes, service centers, car manufacturers, original equipment manufacturers (OEMs), firmware providers, vehicle license offices, cloud storage providers, smart phones and personal computers form a peer-to-peer overlay network. The nodes are divided into multiple clusters where each cluster votes for a single cluster head called overlay block manager (OBM). The OBM nodes manage the public blockchain and verify each incoming transaction. They can get replaced by voting for other ones in case of unreliability or loss of trust. The method of clustering and choosing high resource OBMs form a consortium blockchain decreasing the network overhead significantly. Instead of Proof-Of-Work a scheduled block generation is used eliminating high computational processing power requirements. Each authorized OBM is allowed to generate a new block during a specific time frame. An exemplary clustered network is shown in Figure 2.

Vehicles can interact with the overlay network directly as a member or indirectly through a smart building (e.g. a smart home or service provider). A smart building is managed by a trusted node called Local Block Manager (LBM) which could be integrated in the buildings Internet gateway connecting it to the overlay network. This allows further authorization and privacy features for internal communications but wont be discussed in more detail for the sake of simplicity. Local storage can be used to store privacy-sensitive data that should not get published on the blockchain. With the hash of the data together with a timestamp stored on the blockchain the integrity of the original data can be proofed in retrospect. For example service providers could store client specific data such as the date of the last vehicle maintenance or a repair history log.

Communication is permissioned and can be defined very fine graded. Each OBM holds a list of public key (PK) tuples that declare who is authorized to communicate with each other. When a cluster member wants to grant a new overlay node permission for communication he uploads the corresponding PK to his connected OBM. All transactions are broadcast to all OBMs. When an OBM receives a multisignature transaction with a missing signature it checks the local list of PKs. If it matches one entry the OBM forwards the transaction to the corresponding recipient otherwise it broadcasts the transaction to the other OBMs. However, due to consistent PKs identifiers the architecture design still lacks privacy preserving methods in terms of tracking individual vehicles when location specific data is collected.

3 Use Case and Requirements

This thesis focuses on automotive use cases only, hence, in the following, users and clients are referred exclusively to systems that are integrated in smart vehicles and does not include the owner's smartphone or third party black boxes. The following use cases can be divided into two subcategories having different requirements and purposes.

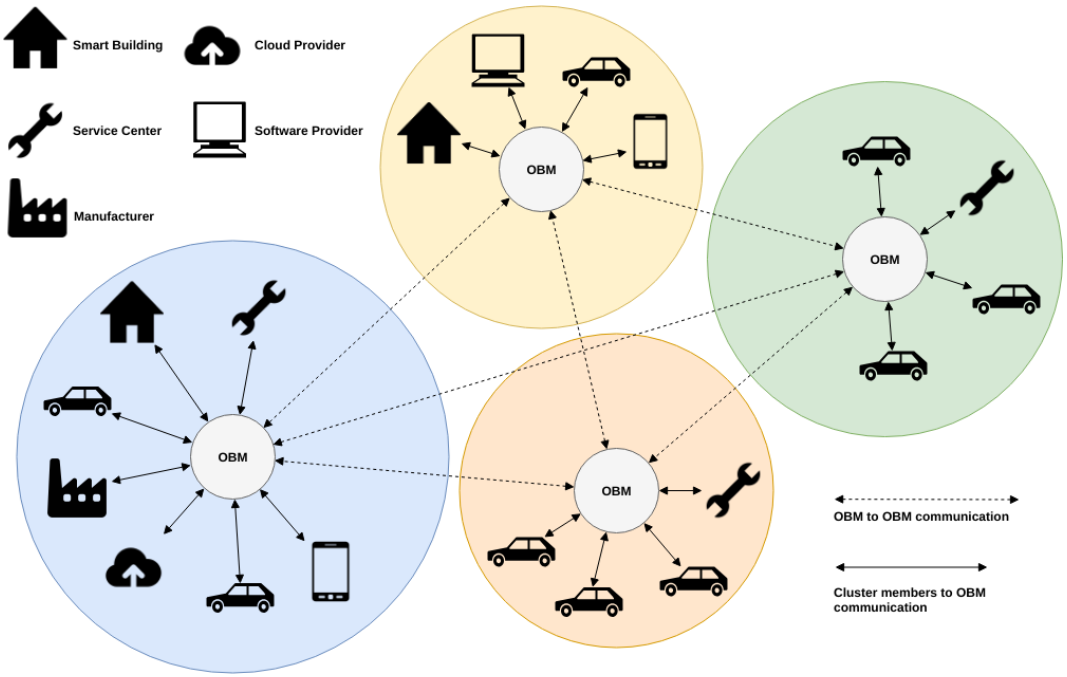

Figure 2: Four clusters connected to each other by trusted OBMs, the authorized validators of the public blockchain. Clients are not bound to specific clusters. They can freely change their cluster minimizing their network delay by performing a soft handover method similar to *Mobile IP*. An OBM could be a car manufacturer or a vehicle license office offering enough resources to handle high traffic volume [Dor+17]. The diagram is adapted from [Dor+17].

3.1 Storing Life Cycles of Vehicles

Storing important life stages of vehicles on the blockchain would provide several advantages. Beginning with the manufacturing of a vehicle, the manufacturer would register it on the blockchain with additional details of e.g. the built-in components and their registration numbers, model specific data and important dates. After that, service providers would publish events such as repair logs, new or changed components, current health of the components and technical inspection certificates. Car dealers would publish or verify resales and the license offices would maintain the licenses and ownerships.

The result would provide a fully transparent history of the registered vehicles, secured by the blockchain's immutable and trustless properties. Thus, buyers could quickly determine the resale price comprehensibly and other third parties (car dealers, service providers, insurances, executive authorities and others) could check licenses and certificates directly on the blockchain which is publicly readable. Since no communication with the other involved parties is required the lookup and validation of information is done instantly and corresponding processes would speed up significantly.

3.2 Utilizing Vehicle Telematics

The second category includes the collection and processing of vehicle sensor data. Instead of third party publishers, road participants themselves publish their sensor data on the blockchain regularly. The sent data can include current speeds, locations, emissions, travel duration, mileage, braking habits and other more specific information. It is important to

note that the user has the full control and can decide what data should be published and in which form it should be presented. In order to stay anonymous for example, specific data can be encrypted and the identity can be hidden.

Storing (or referencing as Section 4 discusses) sensor data on the blockchain enables individual usage-based insurances and general big data applications that are discussed in the following.

Insurances Flexible insurances based on telematics[6] are a popular alternative to traditional insurance plans which are not priced consistently with the individual risks. Such a Pay-How-You-Drive (PHYD) insurance collects a vast variety of sensor data such as mileage, speed, cornering and breaking behavior. The data is analyzed regularly resulting in a driving performance score which provides feedback to the client and influences the premium positively if the score is ranked well. The adoption of usage-based insurances is growing rapidly and multiple studies state that in general, participants indicate positive impacts on their driving style while reducing their premium costs [Hus+15; SW17].

This is especially interesting for young clients who are generally considered to be inexperienced, thus ranked risky, drivers. Traditional insurance plans can get costly whereas the telematics based insurance method is able to classify the skill of the driver and offer matching prices accordingly.

With the development of autonomous vehicles more than the driver's performance becomes relevant. The standard SAE J3016 [Int14] defines six levels of automotive autonomy from no automation at all to full automation. Especially interesting are the levels between the extremes where the vehicles do most of the driving processes. Within these levels, the human driver is still required to take over control in scenarios where the vehicle loses it. Thus, a lower level might require the driver to keep his hands on the steering wheel most of the time during driving. If he does not do so but is involved in an accident one could relieve the autonomous system.

In order to reliable proof or disproof that the user had control over the steering wheel, corresponding sensor data could be published on the blockchain regularly as well. This method is similar to the "dead man's switch", a safety device often placed in trains. Operators need to interact with the device regularly, ensuring that he is well being and can take over the control at any time [Dea].

The use of a blockchain approach offers some advantages. Once the information is published the user cannot alter the data in retrospect despite being the original publisher. Instead of installing a costly black box, insurances read sensor data directly on the blockchain. The identities of the users are kept hidden by default but can be revealed to the chosen insurance company that is authorized to see the data. In case of an accident the data is not stored locally, potentially destroyed. The accident can be resolved faster since all involved parties have instant access to a single view of truth and the insurance companies of all damaged vehicles as well as the curt can act accordingly.

In case of changing the insurance, the deanonymization of the client can be done in retrospect. There is no need to replace a black box and the client can pre-calculate the best fitting insurance plan by letting his history backtested by different service providers.

General Data Analysis By letting users share their sensor data anonymously, third party analysts could evaluate statistics of all kinds in near real time. Mapping services could retrieve the latest data from the blockchain and interpret current traffic situations such

[6] `https://dictionary.cambridge.org/dictionary/english/telematics`

as traffic jams, points of interests or the general congestion of the road network. This is not limited to speed and location data and can be extended for notifications as well. Crowdsourcing a global knowledge pool containing current road workings, accidents, speed cameras, potholes and oil tracks would be applicable. By making use of a voting protocol, other users can agree or disagree with the subject ensuring the validity of the data.

Manipulation can only be done by individual malicious vehicles. The blockchain ensures that each transaction was issued by a registered vehicle even though the identity of the publisher is anonymized (refer to Section 2.3).

Compared to the first category, these use cases are related to Big Data applications which usually are known for huge data throughput. In contrast, the blockchain is known to be rather limiting in regards of throughput and efficiency. This is why benchmarks will test the possibility of a high-performance blockchain and conclude the feasibility of such use cases.

3.3 Observations and Requirements

The term *blockchain* usually refers back to popular currencies like Bitcoin. The fact is that the blockchain might be the technology that powers cryptocurrencies but it is not limited to other use cases. In this case no currency feature is involved which simplifies some of the most important challenges Bitcoin and similar technologies had to solve beforehand. The following observations aim to eliminate misconceptions and to try find reasonable compromises.

3.3.1 The Level of Decentralization

Consider a cryptocurrency (thus holding assets of value, defined in Section 3.3.4) using a consortium blockchain where the nodes that are allowed to mine are preselected and very limited. Less nodes need to validate transactions resulting in less overhead. Much more efficient consensus rules can be chosen eliminating time consuming algorithms such as PoW. This will improve the performance significantly but brings the risk of betrayal of trust (e.g. transactions do not get accepted) or even complete collapse due to new regulations that bans some of the (very limited) nodes in important countries. This is why Ethereum for example is strictly following complete decentralization trying to scale with different approaches like sharding instead of lowering the number of full nodes [Eth].

Having no need for a currency neglects the requirement for complete decentralization. A consortium blockchain without a currency can purely act as a distributed database that is fully transparent and decentralized to a specific point (refer to Section 2.2). A desirable side effect is that transactions can be free by design which is actually one of the main requirements. It would be difficult to achieve with highly decentralized blockchains since Proof-of-Work is costly and Proof-of-Stake consensus rules are not possible without a currency. IOTA[7], a technology using a similar concept to the blockchain is an exception. It has a high level of decentralization and supports free transactions.

3.3.2 The Level of Privacy

A lot of people follow the misconception that Bitcoin is anonymous. The truth is that Bitcoin is only pseudo anonymous meaning that as long as the used Bitcoin address is not linked to an identity there is no way of determining the owner of the address in question. All necessary components in order to interact with the network can get generated locally staying anonymous and the tools to do so are available for free. Nevertheless as soon as a

[7] `https://iota.org`

single transaction can be linked to the owner, all transactions he did before are exposed as well. This is why other cryptocurrencies like Monero and ZCash try to address this issue by using mixing methods or zero knowledge proofs.

The plan of collecting data from vehicles exposing their location faces the same privacy concerns as Bitcoin does. Fortunately no asset is involved, thus addresses do not need to be funded in order to create transactions (similar to zero value transactions). In other words, vehicles can simply generate a new address for each transaction preserving the anonymity and privacy of its owner.

Unfortunately, generating a new throwaway key pair for each transaction makes it difficult to distinguish real vehicles from fake ones. Some kind of authorization proof in the transaction is necessary. This could be done with e.g. none interactive zero knowledge proofs such as zk-snarks [Bit+13] or zk-starks [BS+18]. However, a more efficient and elegant solution would be the use of ring signatures [RST01], a similar concept to group signatures but without a central manager.

3.3.3 Scaleability Solutions and Use Case Limitations

The use cases require scaleability to an extend. With over 40 million cars alone in Germany the blockchain should be able to process at least multiple thousands of transactions per second [Amo]. Fortunately multiple blockchain instances could be deployed. Each specific area of the world could be covered a by a dedicated blockchain instance. A simple, but not necessary best solution would provide each country its own blockchain instance. Other than blockchains with a currency, transactions that refer to other transactions (e.g. during a voting process on traffic events) usually are published at nearby positions. Hence, they are likely to appear in the same blockchain.

In conclusion, even if the blockchain itself is not a bottleneck, other use cases might require a single blockchain or a more complex pegged sidechain architecture in order to support inter-chain transactions. While the conceptual approach should be designed in a way that other use cases are possible, this thesis might not comply with some requirements of these future use cases. However, this does not mean that modifications could be done in retrospect.

3.3.4 Requirements

Scalability The performance correlates heavily with the level of decentralization as Section 3.3.1 observed. Which again, correlates with the ability to scale which results in the speed and throughput of transactions.

User Tracking The identity of participants must stay hidden because other than traditional databases the stored transactions are public which would allow precise user tracking. Transactions with sensitive data should be kept unlinked from other transaction done by the same user and all new transactions should be issued using a new generated address each time. Unfortunately this eliminates systems based on a trust network. For example approaches discussed in Section 2.3 and 2.4 (see [Wan+16; DKJ17]) link user interactions together slowly accumulating a trust factor. This does not respect privacy due to tracking possibilities.

User Authenticity Generating a new address for each new transaction opens the possibility for spamming attacks. Participants could impersonate multiple distinct individuals making false statements and cause traffic jams. This implies the need for a zero knowledge proof of being authenticated without revealing any information of the identity.

Data Validation Transactions might contain incorrect content such as false positive high traffic volumes or wrong location coordinates. Unfortunately it is difficult to reliable filter data that is incorrect by intend or is forged by sensor errors. Especially locations cannot be proofed easily as Section 2.3 discusses.

No Costs Blockchain transactions usually require a fee in order to cover the costs of mining. In this case the user should not face an additional layer of complexity. There should be no costs be involved. In fact, it would probably demotivate users to provide data if they have to pay for it.

Position Perturbation Another important feature would be the ability to disguise the exact position of the vehicle. The reason for that could be as trivial as the fact that anonymous voting is involved and participants should not get identified. In addition it is very difficult to determine the exact GPS coordinates of e.g. a pothole or any object laying on the street without actually being at the exact position.

The Right to be Forgotten (optional) Privacy policies might require a limited storage time for specific data. A blockchain, however, is known for its immutability property which does not allow the deletion of data per se.

Assets (optional) An asset class represents objects of value that can be exchanged between users, often referred to currencies. While transactions should have no fees some use cases might require some kind of asset. This does not apply to data-only use case but might be useful for car sharing services for instance.

Smart Contracts (optional) It would be conducive if the resulting architecture could allow other use cases that might require turing complete applications.

3.4 Choice of the Blockchain Platform

Multiple blockchain technologies that are worth considering are listed in Table 1.

IOTA for example is a promising technology offering free transactions and scalability by adoption staying fully decentralized. However, using their main network would allow traffic participants to spam fake information. Validation would require a significant overhead in comparison to a typical blockchain. Another issue is that it is not possible to determine the correct time order of the transactions which could lead to unprovable false information [Pop17]. At present time IOTA is in a too early state, still facing several critical issues. New promising features such as Qubic are announced but not yet released [Qub]. This is why IOTA is not the final candidate.

The MultiChain platform meets the given requirements and can be easily deployed for prototyping blockchain applications. Since the purpose of the use case is to store data the MultiChain platform seems to be faster and easier to use compared to Hyperledger Fabric or similar projects. MultiChain supports multiple assets at once and can be configured by several parameters (e.g. for eliminating fees). This is why the MultiChain platform will be used as the main component in this thesis. On top of that, MultiChain offers data streams which could simplify the data structures and optimize value retrieval performance.

[8] IOTA's POW should not get confused with traditional ones. It requires much less computational process power.

[9] Ethereum is currently switching to POS.

	MultiChain	IOTA	Ethereum	EOS
Transaction Costs	Yes/No	No	Yes	No
Anonymity by Design	No	No	No	No
Consensus Rule	Permissioned	POW[8]	POW[9]	DPOS
Scalability	Good	Good	Bad	Good
Turing Completeness	No	No	Yes	Yes
Asset Support	if wanted	Yes	Yes	Yes
Accessible	Good	Good; Early stage	Good	Early stage
	Neo	Monero	ZCash	Hyperledger
Transaction Costs	No	Yes	Yes	Yes/No
Anonymity by Design	No	Yes	Yes	No
Consensus Rule	DBFT	POW	POW	Kafka
Scalability	Good	Bad	Bad	Good
Turing Completeness	Yes	No	No	Yes
Asset Support	Yes	Yes	Yes	Yes
Accessible	Early stage	Good	Experimental	Good

Table 1: Overview of possible blockchain candidates with their characteristics at the time being. The "Accessible" property refers to the state of documentation and developer friendliness. Most of them are public blockchains that can be deployed privately as well. However, the deployment of those proofed to be difficult and solutions like MultiChain and Hyperledger are better suited for modified self-deployments.

4 Conceptual Approach

The extended LSB approach discussed in Section 2.4 already provides a good base inspiration how the general architecture will look like. In this conceptual approach, however, the preselected validator nodes (OBMs) are composed exclusively of vehicle license offices and maintain a consortium blockchain. Vehicles are light peers and connect to a OBM nearby.

Privacy and integrity are important factors that need to be addressed. While the information such as the location and the content should be public the identity of the user should be kept hidden. This arises several issues and might require some compromises.

4.1 Challenges and their Solutions

Section 3 introduced several requirements that need to be fulfilled. The following subsections will address these requirements.

Scalability The speed and throughput of transactions are one of the biggest drawbacks of current blockchains. Parameters such as the block size and the block time define a narrow bottleneck for data throughput. Bitcoin for example has a block size of 1MB and a block time of approximately 10 minutes. This limits the transaction rate to about 7 transactions per second which is not sufficient at all for large scale applications such as gathering information of all traffic participants. Increasing the block size is a controversial topic in the cryptocurrency community because it implies more computationally expensive work on the blockchain. Ultimately, in case of fully public blockchains, this will lead to centralization because fewer miners have enough power to handle the workload [Eth]. This is fine for a consortium blockchain since the validator nodes are expected to be powerful servers, maybe a cluster of servers or even supercomputers. Thus, the parameters can be set to a high block size and a low

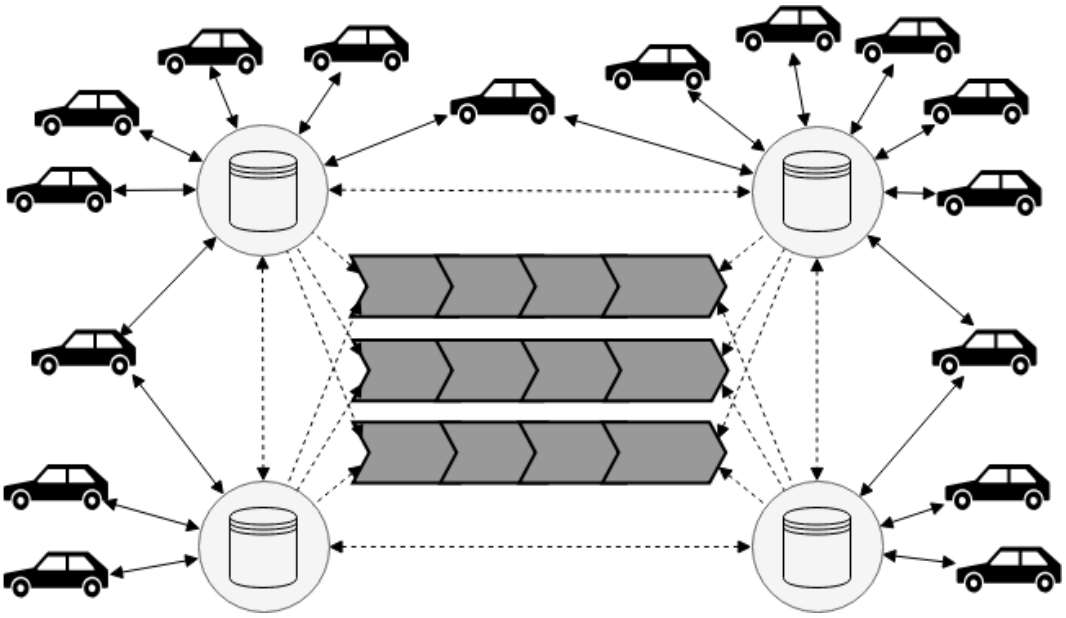

Figure 3: Car manufacturers and mapping service providers are connected in a peer to peer network. They are the validator nodes maintaining the blockchain instances (colored in gray). Vehicles connected to any of the validator nodes can publish transactions.

block time. Furthermore the use case allows horizontal scaling by using more than one blockchain instance. This is explained in more detail in Section 3.3.3 and 4.3.2.

User Authenticity Generating a new address for each new transaction opens the possibility for spamming attacks. Participants could impersonate multiple distinct individuals, making false statements and cause traffic jams. This implies the need for a zero knowledge proof of being authenticated without revealing any information of the identity. Such a proof of membership is discussed in Section 4.2.

Data Validation A voting protocol will give new information a weight. Vehicles nearby will vote, agreeing or disagreeing that the information is correct or incorrect. This also allows to mark events as expired if e.g. an accident is already resolved and not relevant anymore.

The Right to be Forgotten Data on the blockchain enjoy life-time immutability. A modification of a transaction in form of overwriting or deletion would make the corresponding block invalid, transitively invalidating all successor blocks. By storing the data off-chain, immutability would be preserved if the data itself is only referenced by a unique hash. Deleting the data would not change anything on the blockchain, the reference simply could not be resolved anymore.

4.2 Privacy Preserving Methods

Depending on the use case there are two different scenarios. Sometimes clients need to authenticate their true identity, then the service provider needs to know the identity of the client. The solution is rather simple. Service providers publish their PKs on the blockchain which can be used by clients to encipher their identity. After all, only the service provider is able to authenticate the client.

However the more interesting scenario is when clients need to stay anonymous but also need to authorize themselves which is the case when traffic statistics are published on the blockchain. Then, clients need to provide a proof that they are registered vehicles and do not belong to a virtual fake fleet. This can be done by using ring signatures that allow a proof of membership [RST01]. Initially all vehicles are registered on the blockchain identified by their permanent PK. This means, that these PKs are verified by the OBMs to be real vehicles. The corresponding private keys are stored locally in a trusted execution environment (TEE). Each time when a client creates an anonymous transaction he generates a new throwaway key pair which will be used for the sender address. Within the transaction he provides a ring signature together with a list of PKs that were used to generate the signature. The list contains his own permanent PK and a sufficient amount of random PKs that are registered on the blockchain. This method allows blockchain nodes to validate registered participants but they are not able to determine the real signer which preserves the privacy of the users. Unfortunately it also introduces some issues.

Privacy leaks The ring signatures make use of a relatively small subset of PKs. The more PKs are used, the higher the level of privacy can be ensured. The probability of guessing the correct signer decreases linearly, proportional with the size of the PK pool. However, the size of the signature and the required computational power grows linearly as well which is why only a relatively small number of PK is used.

Because transactions are linked to locations, privacy could be leaked if the algorithm that chooses the used PKs is acting completely random. As an example consider the following scenario. A user drives along a route from the starting point A over points B and C ending at D. At each point A-D he sends location based data to the blockchain and uses completely random foreign registered PKs to generate the ring signatures.

Case 1: For each point A, B, C and D he chooses different random PKs. Then, all ring signatures done by the user will have one common PK, namely the one of the user. An attacker could search for recurring PKs in a specific area allowing him to find out the route of the user, thus, he would able to track him.

Case 2: For each point A, B, C and D he chooses different random PKs but the choice is limited to those PKs that were used nearby lately. The attacker would not be able to search for recurring PKs anymore but regions with a low population density will still suffer from statistical analysis attacks. Thus, the level of privacy would not only correlated with the amount of PKs but also with the density of location related participants.

Case 3: For the complete route he chooses the same list of PKs for all used ring signatures. Similar to the second case the chosen PKs are location related. Then, an attacker would be able to track the route with the unique signature but is not able to find out which exact user is behind the signature.

Privacy could be leaked as well if it is possible to detect users based on their driving habits. This could be the case if, for instance, an user drives on a highway with a unique constant speed.

Voting Protocols As described in Section 4.1, it is difficult to determine the correctness of information. This is why voting protocols should enable other road participants to agree or disagree with information, giving it a meaningful value. As usual, in a voting process, only registered users should be able to vote and each participant may only vote once. Ring signatures, however, allow users to hide behind a group of users, thus, double voting multiple times would be applicable.

An extension of ring signature could solve these issues. Linkable ring signatures add the ability to externally determine if two or more ring signatures were signed by the same

individual [TW05]. Voting would be possible again and together with the method of the second case, the privacy of the user would be preserved if enough other road participants are nearby.

However, using ring signatures still risks of having privacy leaks, especially in regions with a low density population. The amount of PKs need to be sufficient and the list should be refreshed regularly otherwise statistical analysis attacks would be practicable. Other methods that could enhance the level of privacy like the usage of zk-snarks and zk-starks proofs will be discussed and reviewed.

4.3 Optimizations

Transactions on the blockchain carry a significant overhead compared to entries in traditional distributed database solutions. Note that mainly IoT devices[10] are involved which increases the need for efficiency in regards of processing work and space requirements. For example, each transaction contains a signature of a specific length which makes the choice of the underlying cryptography algorithms critical.

4.3.1 The Choice of Cryptographic Systems and Signature Schemes

If, for instance, a Rivest–Shamir–Adleman (RSA) cryptosystem is used the key sizes need to be at least 3072 bits long[11]. As comparison, the Elliptic Curve Cryptography (ECC) system offers security equivalent to RSA with a much smaller required key size of about 283 bits. As a result, the key generation performance also differs significantly. Previous works such as [Jan04] showed that ECC outperforms RSA by a factor of over about 30 which can be explained by the fact that the key generation time grows exponentially for RSA but linearly for ECC. The corresponding signatures yield the same factors regarding the size [Jan04]. A naive implementation of ring signatures lists all used PKs together with some values that are required. Because this list represents the signature, ring signatures grow linearly in size by the amount of members. This also means that ECC would allow much smaller ring signatures compared to RSA versions.

Another interesting signature scheme are so called *Schnorr signatures* which ultimately could reduce the signature size to its minimum by making use of its support for key aggregation [Sch91; Max+18]. Applicability will be investigated and discussed.

4.3.2 Improvements with GeoHashes

Instead of using traditional GPS coordinates the GeoHash format offers some useful properties and will be used to store geographical positions. GeoHash is a public domain algorithm written by Gustavo Niemeyer. It encodes geographic coordinates into a unique string that allows arbitrary precision which means that cutting off a suffix from the GeoHash string results in a less accurate information about the location. For example a GeoHash with the length of two characters will describe an area with a diameter of up to 630 km. The exact distances depending of the string length up to 8 characters are listed in Table 2. This although means that similar prefixes usually imply nearby places where the length of the shared prefix defines the proximity [Geoa]. Figure 4 visualizes some areas using GeoHash strings of different sizes.

By using GeoHashes the world map can be split into multiple sections, each section maintained by its own blockchain. This results in multiple blockchain shards allowing more

[10] Smart vehicles can be considered to be IOT devices albeit being relatively powerful.

[11] According to https://www.keylength.com/en/3/

length	latitude diff	longitude diff	km diff
1	23	23	2500
2	2.8	5.6	630
3	0.70	0.70	78
4	0.087	0.18	20
5	0.022	0.022	2.4
6	0.0027	0.0055	0.61
7	0.00068	0.00068	0.076
8	0.000085	0.00017	0.019

Table 2: The length of a GeoHash string determines the precision which results in high deviations for short GeoHashes [Geoa].

Figure 4: A screenshot taken from the GeohashExplorer using Google Maps ([Geob; Goo]). The world map is divided into different unique strings where each string represents a specific area of the world in form of a rectangle. Increasing the length of the string results in more precise smaller areas. Germany for example is fully covered by the GeoHash prefixes `u0`-`u3`.

throughput overall by scaling horizontally. Each blockchain (identified by a GeoHash prefix) will contain several streams where clients can send their data to. Each stream (again identified by a GeoHash affix) stores items consisting of a key-value pair. The key describes an event id that is time specific and the value will hold the transaction. Publishing a new transaction with the same item id will update it. For example a new transaction notifies about an

accident that occurred at the coordinates 50.780777 and 6.108539 (corresponds to the latitude and longitude values) which converts to the GeoHash `u1h2gh257qby`. The transaction would be sent to the blockchain identified by `u1` and published in the stream identified by `h2`. The transaction itself would just contain the GeoHash suffix up to `gh257qby` according to the desired precision. While this method does not only allow data compression it although improves the performance by using specific data structures to store the transactions.

5 Implementation

One goal of the thesis is a working proof of concept implementation. However, some aspects might be simplified so that the complexity stays in scope of the subject. The following subsections will list the order of implementation steps that depend on each other, discuss possible issues and define the exactness of the implementation. A high level diagram of the whole project is shown in Figure 5.

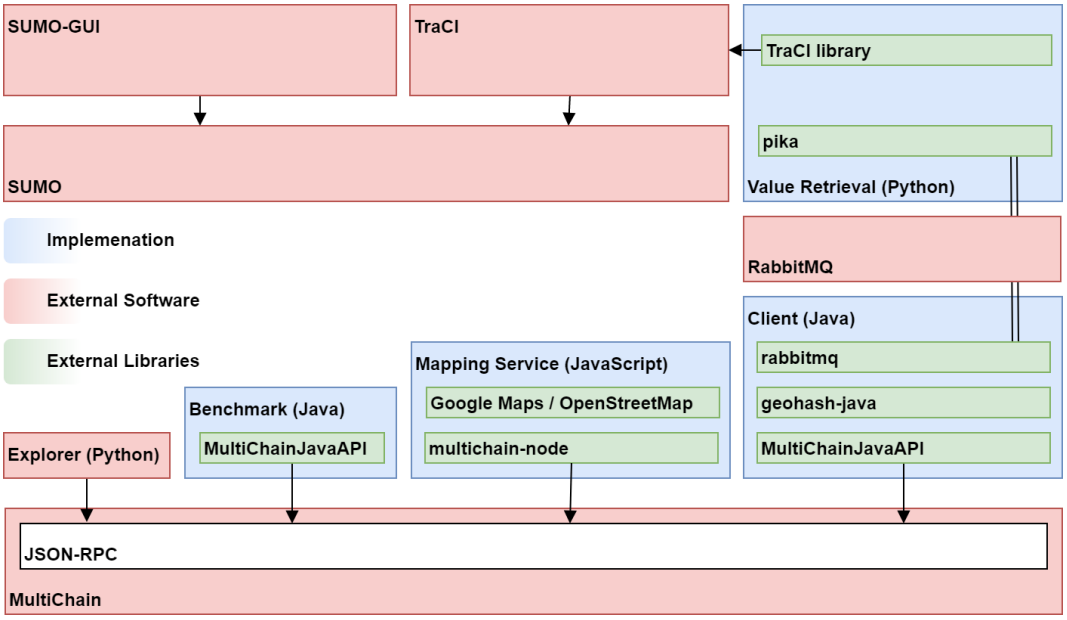

Figure 5: A high level diagram showing all used components that form the project. The SUMO simulation is strictly isolated from the blockchain part by using a message broker. The broker does not only provide a simple interface between different programming languages but also allow to process transactions simultaneously if necessary.

5.1 MultiChain Platform

Decentralization usually slows down the process of data. Therefore the performance of the MultiChain platform will be tested. The goal is to find out how many transactions per seconds are possible using just a single blockchain instance. The benchmark will run on a single machine. If it continues to keep up with the amount of transactions the benchmark setup will be upgraded with multiple computers all connected in a fast local network. This should reveal possible bottlenecks, an important factor to determine the theoretical amount of instances that are required to cover the whole world map.

Programming Language The benchmark will be implemented using the Java wrapper[12] for the MultiChain JSON-RPC interface.

Variables of Interest Plots should reveal possible bottlenecks by testing the following variables against each other.

- The amount of nodes against the relative amount of transactions that actually get confirmed
- The size of the payload against the relative amount of transactions that actually get confirmed
- The amount of concurrent transactions per second against the amount of confirmations per second

Possible Issues While performing some tests in beforehand the MultiChainJavaAPI library showed that it misses some API calls and suffers from several bugs especially when working with raw transactions. These bugs are critical and need to be fixed. Furthermore the library does a lot of data serialization which could slow down the process. Simple HTTP requests directly communicating with the interface using e.g. "curl" might be more efficient. Another issue of the library is its poor TCP connection handling resulting in errors due to too many open connections while benchmarking. However, using multiple spamming instances should solve possible bottlenecks regarding the processing overhead.

Another issue arises from the MultiChain platform itself. It does not offer a method to determine the total number of transactions other than accessing the underlying database of a full node. However, it is possible to retrieve the number of transactions of a specific wallet which will be taken advantage of. Simple benchmark tests showed that the number of transactions does not rise monotonously which should be the case since the blockchain is an append-only ledger. Instead, the number decreases randomly before growing significantly again which makes it difficult to interpret benchmarking results. The developers of MultiChain investigated this behavior and came to the conclusion that some transactions are count twice which results in incorrect values. This is likely to be fixed in a future release. Until then, a workaround must be found.

5.2 Architecture and Protocols

The next step will extend the first one which already provides some base classes and implement the general architecture with its protocols.

Libraries The MultiChainJavaAPI and geohash-java[13] will be used.

Issues There is no highly tested library for ring signatures. The implementation might have to be done from scratch or an existing approach need to be modified.

Simplifications MultiChain offers the possibility to create Bitcoin like full nodes where third party light wallets are able to connect to. Unfortunately this does not work for MultiChain specific features such as streams which this project makes heavily use of. Since implementing a light wallet would go beyond the scope of this thesis it will be simulated instead. A Java class will offer some light wallet specific methods but behind the scenes the application communicates with a full node having root access.

[12] `https://github.com/SimplyUb/MultiChainJavaAPI`
[13] `https://github.com/kungfoo/geohash-java`

5.3 Architecture Benchmark

Similar to the first step the performance of the architecture and protocols will be tested. The result will predict the theoretical processing time of a single transaction from it's creation to it's publication taking into account various limiting factors such as the block time, block size, key generation and network latency. The results will be plotted and discussed. This also should give an idea if such an architecture is sustainable and possible for real implementations.

5.4 Visualization and Evaluation

To evaluate and test the functionality of the project a data set with test cases returning comprehensible results is needed. Predefined scenarios are difficult to create and do not reflect real world situations. This is why an existing traffic simulation framework will be used instead. Another website will expose behind the scenes blockchain specific data listing all transactions in raw.

Figure 6: The graphical user interface showing a detail view of the map of Aachen. The road network is taken from OpenStreetMap (openstreetmap.org) and converted into a configuration file that is compatible with the *SUMO* application. Vehicles are set to spawn randomly colored in yellow. The result allows real world like scenarios on real road infrastructures.

The *SUMO* package[14] (**S**imulation of **U**rban **MO**bility) is an open source road traffic simulation which can handle large road networks. It allows to spawn vehicles in predefined locations following a specific route. Placing blockades could represent accidents and blocking (or speed limiting) lanes could represent road construction work. It also offers a traffic control interface (*TraCI*) which allows retrieving current values (speeds, emissions, locations, ...) and interactions with the simulation. A graphical user interface (included in the *SUMO* package) visualizes the simulation in a real time map. A possible scenario is shown in Figure 6.

Programming Language *TraCI* is available in form of libraries written in various programming languages. The developers of *SUMO* highly suggest using the Python library since it supports all *TraCI* commands [Sum]. This is why Python will be used to retrieve simulated vehicle data. The second website which lists all transactions in plain

[14] `http://sumo.dlr.de`

text wont be written from scratch. Existing projects like the MultiChain Explorer[15] might be already a good choice. Since it is open sourced using the GNU AFFERO GENERAL PUBLIC license it can be forked and customized.

Possible Issues The *TraCI* library slows down the simulation significantly depending on how many *TraCI* computations are done during each simulation step [Sum]. Considering that sensor data of all vehicles are monitored it is very likely that the simulation is not able to show real time traffic anymore. This could be avoided by monitoring only specific road sections that are interesting. However, the performance of the simulation also depends on the size of the road network and the amount of vehicles. Finding the right parameters should give promising results.

Using the Python library rises the level of effort needed to publish transactions on the blockchain since it needs to be implemented all over again. Using existing Python libraries will be possible but using an asynchronous task queue might be a better solution. The existing Java implementation will work off the task queue while lowering the amount of processing power during each simulation step.

5.4.1 Evaluation

The goal of the thesis is accomplished if the transactions on the blockchain reflect the traffic information on the simulated map. Sensor data and statistics of e.g. current traffic should be retrieved exclusively from the data stored on the blockchain. The used protocols should be implemented following privacy preserving concepts as well as the latest security conventions. Ultimately, the user should be able to use the service as before without even knowing that the blockchain is used in the backend. Common issues such as poor performance and lack of anonymity should be discussed and prevented as well as possible. Drawbacks, issues and compromises should be disclosed.

6 Project plan

The following project time plan is created for a time range of approximately 4 months. All steps include the writing of the corresponding sections.

- **Implementation of base classes and first benchmark (1 week):** This also includes fixing bugs in the used Java library.
- **Completion and optimization of the draft (architecture and protocols) (1 week):** The conceptual definition still lacks formalities.
- **Implementation of the simulation (3 weeks):** This will take some time especially when ring signatures need to be implemented.
- **Writing phase (2 weeks)**
- **Benchmark of the architecture (1 week)**
- **Finishing the writing work (5 weeks)**
- **Designing the presentation and buffer (1-3 week)**

[15] `https://github.com/MultiChain/multichain-explorer`

Abbreviations

dPOS Delegated Proof-Of-Stake

ECC Elliptic Curve Cryptography

GPS Global Positioning System

IoT Internet of Things

LBM Local Block Manager

LSB Lightweight Scalable Blockchain

MVCC Multiversion Concurrency Control

OBM overlay block manager

OEM original equipment manufacturer

PK public key

PoS Proof-Of-Stake

PoW Proof-Of-Work

RSA Rivest–Shamir–Adleman

SSID Service Set Identifier

TEE trusted execution environment

TOTP Time-based One-Time Password algorithm

References

[Dea] *Automated monitoring.* URL: `https://www.gpo.gov/fdsys/pkg/CFR-2004-title49-vol4/xml/CFR-2004-title49-vol4-sec238-237.xml` (visited on 08/02/2018).

[Blo] *Avoiding the pointless blockchain project | MultiChain.* URL: `multichain.com/blog/2015/11/avoiding-pointless-blockchain-project/` (visited on 07/27/2018).

[BS+18] Eli Ben-Sasson et al. *Scalable, transparent, and post-quantum secure computational integrity.* Cryptology ePrint Archive, Report 2018/046. `eprint.iacr.org/2018/046`. 2018.

[Bit+13] Nir Bitansky et al. "Succinct Non-interactive Arguments via Linear Interactive Proofs". In: *Theory of Cryptography.* Ed. by Amit Sahai. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 315–333. ISBN: 978-3-642-36594-2.

[Cor18] Foamspace Corp. "FOAM Whitepaper". In: 2018. URL: `foam.space/publicAssets/FOAM_Whitepaper_May2018.pdf`.

[DKJ17] Ali Dorri, Salil S. Kanhere, and Raja Jurdak. "Towards an Optimized BlockChain for IoT". In: *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation.* IoTDI '17. Pittsburgh, PA, USA: ACM, 2017, pp. 173–178. ISBN: 978-1-4503-4966-6. DOI: `10.1145/3054977.3055003`. URL: `doi.acm.org/10.1145/3054977.3055003`.

[Dor+17] Ali Dorri et al. “BlockChain: A distributed solution to automotive security and privacy”. In: *CoRR* abs/1704.00073 (2017). arXiv: 1704.00073. URL: arxiv.org/abs/1704.00073.

[EN10] Ramez Elmasri and Shamkant Navathe. *Fundamentals of Database Systems*. 6th. USA: Addison-Wesley Publishing Company, 2010. ISBN: 0136086209, 9780136086208.

[Geoa] *Geohash - Wikipedia*. URL: en.wikipedia.org/wiki/Geohash (visited on 03/20/2018).

[Geob] *GeohashExplorer*. URL: geohash.gofreerange.com/ (visited on 05/30/2018).

[Ger+14] M. Gerla et al. “Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds”. In: *2014 IEEE World Forum on Internet of Things (WF-IoT)*. 2014, pp. 241–246. DOI: 10.1109/WF-IoT.2014.6803166.

[Goo] *Google Maps*. URL: google.de/maps/@50.786419,6.0833356,15z (visited on 05/30/2018).

[Hus+15] Siniša Husnjak et al. “Telematics System in Usage Based Motor Insurance”. In: *Procedia Engineering* 100 (2015). 25th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2014, pp. 816 –825. ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2015.01.436. URL: http://www.sciencedirect.com/science/article/pii/S1877705815004634.

[Int14] SAE International. *AUTOMATED DRIVING LEVELS OF DRIVING AUTOMATION ARE DEFINED IN NEW SAE INTERNATIONAL STANDARD J3016*. 2014.

[Jan04] Nicholas Jansma. “Performance comparison of elliptic curve and rsa digital signatures”. In: May 2004.

[Jes13] Tobias Jeske. “Floating Car Data from Smartphones: What Google and Waze Know About You and How Hackers Can Control Traffic”. In: 2013.

[Len+08] Vincent Lenders et al. “Location-based Trust for Mobile User-generated Content: Applications, Challenges and Implementations”. In: *Proceedings of the 9th Workshop on Mobile Computing Systems and Applications*. HotMobile ’08. Napa Valley, California: ACM, 2008, pp. 60–64. ISBN: 978-1-60558-118-7. DOI: 10.1145/1411759.1411775. URL: doi.acm.org/10.1145/1411759.1411775.

[Max+18] Gregory Maxwell et al. *Simple Schnorr Multi-Signatures with Applications to Bitcoin*. Cryptology ePrint Archive, Report 2018/068. https://eprint.iacr.org/2018/068. 2018.

[Mul] *MultiChain | Open source blockchain platform*. URL: multichain.com (visited on 06/01/2018).

[Pop17] Serguei Popov. “On the timestamps in the tangle”. In: 2017.

[Inr] *Privacy Policy | INRIX*. URL: http://inrix.com/site-privacy-policy (visited on 07/06/2018).

[Qub] *Qubic: Quorum-based Computations - Powered by IOTA*. URL: qubic.iota.org/ (visited on 06/03/2018).

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. “How to Leak a Secret”. In: *Advances in Cryptology — ASIACRYPT 2001*. Ed. by Colin Boyd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 552–565. ISBN: 978-3-540-45682-7.

[Sch91] C. P. Schnorr. “Efficient signature generation by smart cards”. In: *Journal of Cryptology* 4.3 (1991), pp. 161–174. ISSN: 1432-1378. DOI: 10.1007/BF00196725. URL: https://doi.org/10.1007/BF00196725.

[Eth] *Sharding FAQ · ethereum/wiki Wiki*. URL: github.com/ethereum/wiki/wiki/Sharding-FAQ (visited on 06/01/2018).

[SW17] Miremad Soleymanian and Charles B. Weinberg. "Sensor Data , Privacy , and Behavioral Tracking : Does Usage-Based Auto Insurance Benefit Drivers ?" In: 2017.

[Sum] *Sumo*. URL: sumo.dlr.de/wiki/TraCI (visited on 03/01/2018).

[Tom] *TomTom Privacy | Drive*. URL: tomtom.com/en_us/privacy/drive/ (visited on 07/05/2018).

[TW05] Patrick P. Tsang and Victor K. Wei. "Short Linkable Ring Signatures for E-Voting, E-Cash and Attestation". In: *Information Security Practice and Experience*. Ed. by Robert H. Deng et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 48–60. ISBN: 978-3-540-31979-5.

[Wan+16] Gang Wang et al. "Defending Against Sybil Devices in Crowdsourced Mapping Services". In: *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. MobiSys '16. Singapore, Singapore: ACM, 2016, pp. 179–191. ISBN: 978-1-4503-4269-8. DOI: 10.1145/2906388.2906420. URL: doi.acm.org/10.1145/2906388.2906420.

[Amo] *Wie viele Autos gibt es in Deutschland? | markt.de*. URL: markt.de/ratgeber/autos/wie-viele-autos-gibt-es-in-deutschland/ (visited on 05/02/2017).

Attributions

Figures in this paper make use of several third party icons that are attributed in the following.

- *car, citroen, top, vehicle icon* by Benjamin STAWARZ on iconfinder.com is licensed under CC BY 3.0 / Colorized from its original
- *Alert, danger, warn, warning icon* by Rawnly on iconfinder.com is licensed under CC BY 2.5
- *Factory icon* by WPZOOM on iconfinder.com is licensed under CC BY-SA 3.0
- *Map icon* by Cole Bemis on iconfinder.com is licensed under CC BY 3.0
- *Home, house, map, place icon* by strongicon on iconfinder.com is licensed under CC BY 3.0
- *Configuration, fix, repair, setting, setup, tool icon* by Chanut is Industries on iconfinder.com is licensed under CC BY-SA 3.0
- *Cloud, storage icon* by Stephen Hutchings on iconfinder.com is licensed under CC BY-SA 3.0
- *Wifi icon* by Ionicons on iconfinder.com is licensed under MIT License
- *Package icon* by Github on iconfinder.com is licensed under MIT License
- *Wifi icon* by Andreas Larsen on iconfinder.com is free for commercial use
- *Communication, mobile, phone icon* by First Styles on iconfinder.com is licensed under CC BY 3.0