Leveraging Distributed Ledger Technology for Decentralized Mobility-as-a-Service Ticket Systems

Marc Leinweber (KASTEL, Karlsruhe Institute of Technology (KIT)), *Niclas Kannengießer (AIFB, Karlsruhe Institute of Technology (KIT))*, *Hannes Hartenstein (KASTEL, Karlsruhe Institute of Technology (KIT))*, *Ali Sunyaev (AIFB, Karlsruhe Institute of Technology (KIT))*

Content

This work was supported by funding from the topic Engineering Secure Systems of the Helmholtz Association (HGF) and by KASTEL Security Research Labs. We thank Adriano Castro and Tilo Spannagel for the implementation of the benchmarking framework.

This version is a preprint of our work presented at “14. Wissenschaftsforum Mobilität 2022” to be published at Springer.

1 Introduction

Mobility-as-a-Service (MaaS) aims at increasing sustainability of mobility as well as improving convenience and flexibility for customers when using public mobility services (e.g., car sharing services, railroad services), and tailoring tariffs for using such services (Jittrapirom et al., 2017; Nguyen et al., 2019; Preece & Easton, 2019). To approach these MaaS goals, mobility providers can form ticket federations that offer mobility services via joint ticket systems. Customers can purchase tickets for travels, which can involve mobility services offered by any mobility provider in the federation, via such an MaaS ticket system with a single account. In this way, convenience and flexibility for customers in using public transportation can be increased.

MaaS ticket systems are usually offered by organizations with large market power (Nguyen et al., 2019). Such organizations often take key roles in IT governance of MaaS ticket systems, which can make ticket federations dependent on such central organizations (Farsi et al., 2007). Small mobility providers usually do not possess sufficient rights to participate in decisions on operation fees and the inclusion of mobility providers (Bundeskartellamt, 2016, 2019, 2022). Dependencies of ticket federations on central organizations may make it easier for these organizations to exploit the federation (e.g., by taking high fees from other mobility providers for using the ticket system) and to exclude mobility providers that compete with their own service offerings. Exclusion of mobility providers limits the range of mobility services available to customers and hinders the attainment of MaaS goals in terms of convenience and flexibility in the use of these services.

To decrease such dependencies, MaaS ticket systems can be decentralized. In decentralized ticket systems, accountabilities and efforts related to the system (e.g., for system operation) are distributed across mobility providers that are largely independent from each other (Li et al., 2020; Nguyen et al., 2019) so that each provider can become involved in the development, operation, and governance of the MaaS ticket system (Beck et al., 2018). In return to efforts put into the ticket system, for example, related to development and operation, all mobility providers in a ticket federation can obtain decision rights so that all providers can jointly govern the MaaS ticket system (Beck et al., 2018), for example, by voting on the joining of new providers. Through decentralization, federations can become less dependent on central organizations, which may make it easier for new providers to join a federation and offer their services. In this way, decentralization of ticket systems, especially of the ledgers recording ticket purchases in ticket systems, can support the attainment of MaaS goals.

Distributed ledger technology (DLT) seems to be suitable for the implementation and operation of information systems with decentralized governance (Beck et al., 2018; Heines et al., 2021), including MaaS ticket systems (Li et al., 2020; Nguyen et al., 2019; Preece & Easton, 2019). DLT enables the operation of distributed ledgers in a decentralized manner and builds on the concept of replicated state machines (RSMs; Schneider, 1990). In the RSM concept, each node maintains a consistent state, which is, for example, specified by the transactions stored in the ledger of a ticket system. To achieve consistency, states are synchronized across all nodes in the system by a consensus mechanism, such as Practical Byzantine Fault Tolerance (Castro & Liskov, 1999). Because nodes in DLT systems may be arbitrarily unreachable (e.g., due to a crash) and participants in the system (e.g., customers, mobility providers) may even issue malicious transactions (e.g., by equivocation), consensus mechanisms used in DLT systems are usually robust to tolerate such faults (Pease et al., 1980), increasing the overall availability and reliability of a system (Gupta, 2016). Such robust consensus mechanisms often resemble voting processes and enable decentralized governance by DLT systems on behalf of their participants. By using DLT, each mobility provider in a ticket federation can operate their own node with a replication of the ledger used in the ticket system.

The extensive ledger replication increases system availability, can avoid information asymmetries, and can decrease dependencies on central organizations (e.g., in terms of system operation).

Despite the potential of DLT to support the attainment of MaaS goals, especially the replication process and consensus finding in DLT systems lead to challenges in fulfilling ticket system requirements, for example, related to confidentiality (Cheng et al., 2019; Nguyen et al., 2019; Wang & Zhang, 2021) and transaction processing (Dinh et al., 2017; Preece & Easton, 2019). For example, synchronization overhead in consensus finding can elongate transaction processing (e.g., approx. one hour for transaction confirmation in the Bitcoin system) and, thus, can render DLT unsuitable for MaaS. Moreover, the extensive replication of the ledger across all nodes in a ticket system makes travel data accessible for each mobility provider operating a node, which can compromise travel data confidentiality. To understand to what extent DLT-based MaaS ticket systems can meet requirements of MaaS in the presence of those challenges, we approach the following research question: *How can DLT be used to decentralize ticket systems to meet MaaS requirements in a real-world scenario?*

As illustrated in Fig. 1, we first introduce an MaaS scenario for public transportation and compile key MaaS goals as well as requirements for MaaS ticket systems to design a prototypical MaaS ticket system. In our MaaS scenario, we describe why decentralization can contribute to the attainment of MaaS goals and explain how DLT can support the fulfillment of requirements for decentralized MaaS ticket systems. Next, we collate challenges of using DLT in ticket systems and outline possible design options, such as the use of oracles or trusted execution environments (TEEs), to tackle these challenges. We briefly compare the design options to identify benefits and drawbacks with respect to the requirements of decentralized ticket systems. Under consideration of the identified benefits and drawbacks of individual design options and the requirements for MaaS ticket systems, we propose a prototypical design of a decentralized MaaS ticket system based on DLT and TEEs for public transportation. We preliminarily evaluate the prototypical system to obtain initial insights into its practicality. For the evaluation, we developed an initial benchmarking framework that is based on parallel clients to model real-world MaaS ticket systems. Last, we discuss our findings and outline future research directions regarding the use of TEEs in decentralized MaaS ticket systems.

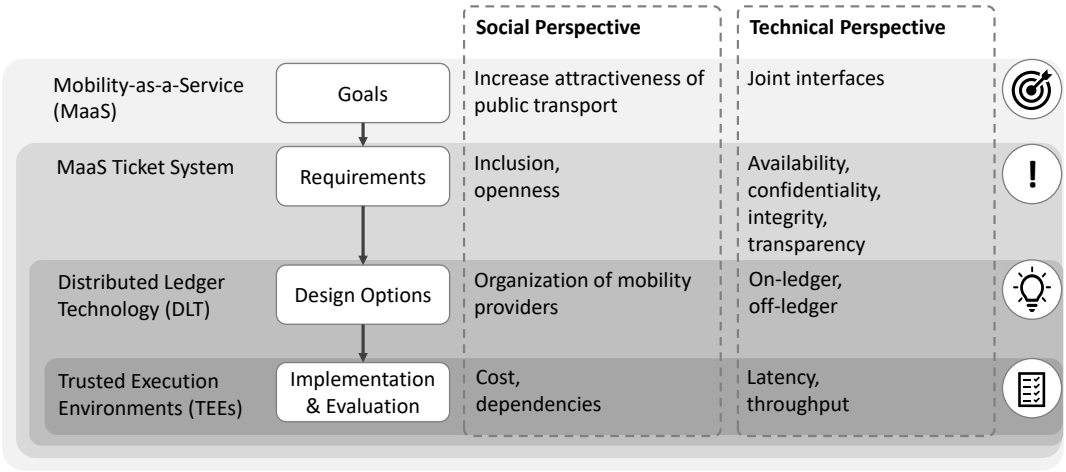

Fig. 1: Overview of the structure of this work. Based on the MaaS goals and corresponding requirements for MaaS ticket systems, we discuss design options for DLT-based MaaS ticket systems. We choose TEEs to address challenges of DLT systems and propose an MaaS ticket system prototype. We preliminarily evaluate the system focusing on throughput and transaction processing time.

2 Distributed Ledger Technology in Mobility-as-a-Service Ticket Systems

2.1 Our Mobility-as-a-Service Scenario and Its Requirements for Ticket Systems

We focus on an MaaS scenario with a ticket federation for public transport, including trains, trams, and busses. The federation offers pay-as-you-go tariffs to increase convenience and flexibility of using mobility services for customers and to offer tailored fares for using mobility services as targeted in MaaS. In pay-as-you-go tariffs, fares for the actual usage of mobility services are calculated based on proofs-of-interaction (Jittrapirom et al., 2017). Proofs-of-interaction are check-ins and corresponding check-outs: Customers check in when entering vehicles and check out when leaving vehicles of mobility providers. Customers are only authorized to use mobility services after they checked-in successfully. All proofs-of-interaction are recorded by the ticket system. To guarantee utility of the ticket system, transactions related to proofs-of-interaction must be processed within very short time (i.e., starting from the issuance of a transaction from the vehicle until its final inclusion into the ledger). Because the proofs-of-interaction are used to calculate fares and to verify the authorized use of mobility services by customers, the ticket system must be highly available and integrity of stored data must be ensured.

To legally frame the use and provision of mobility services, customers in the ticket system establish legal relationships with a single mobility provider in the ticket federation. Mobility providers are responsible for billing, payment claims, and support requests. Customers are responsible for paying bills via conventional payment options, such as credit cards. To anticipate privacy infringements and protect business secrets of mobility providers (Lamberti et al., 2019; Nguyen et al., 2019; Preece & Easton, 2019) while allowing for the calculation of fares per customer and corresponding revenues per mobility provider, only selected employees (e.g., billing staff) of the contract partner of each customer must be able to associate proofs-of-interaction with customers. Other entities in the system must not be able to link histories of proofs-of-interaction to customers. Ticket inspectors must be able to verify the authorized use of a mobility service by a customer based on the current check-in but must not be able to learn customers' past travels (Nguyen et al., 2019).

A ticket federation agrees on a billing function and a billing interval in negotiations between mobility providers. The billing function includes a pricing model for all mobility services offered in the ticket federation and calculates travel fares per customer based on their stored proofs-of-interaction at the end of a specific billing interval (e.g., once a month). Each mobility provider collects all fares to be paid by its contract partners (i.e., customers), including those fares that accrue for using mobility services of other providers. After a mobility provider received payments from a customer, the mobility provider distributes revenues across all federation members according to the usage of their mobility services. For billing and payment operations, selected employees of mobility providers must be able to calculate total fares and revenue distribution based on the proofs-of-interaction of customers with whom they have legal relationships. Mobility providers that receive payments from another provider in revenue distribution must be able to verify the balance of their revenues at any time. Customers must be able to obtain an overview of their fares. To avoid fraud related to billing, data stored in the ledger must only be mutable if customers and mobility providers associated with that data agree on the manipulation.

2.2 Decentralization of Mobility-as-a-Service Ticket Systems

MaaS ticket systems are information systems that are comprised of a social subsystem, which includes a variety of social actors (e.g., customers, mobility providers), and a technical subsystem, which includes devices, algorithms, and tools (e.g., billing processes, vehicles), as well as their relationships and interactions (Chatterjee et al., 2021). Information systems are embedded in a political environment, which includes societal norms, laws, and

regulations. For the reasonable decentralization of MaaS ticket systems, political, socials, and technical MaaS goals and associated requirements for ticket systems need to be considered. A political MaaS goal is to reduce carbon dioxide emissions by making public transportation more attractive for individuals so that the usage of private transportation decreases (European Environment Agency, 2021; Gould et al., 2015). To achieve that goal, the social subsystem aims at making mobility service more convenient for individuals and at increasing flexibility in choosing mobility services. Moreover, fares should be tailored to customers' uses of mobility services (Jittrapirom et al., 2017). Market entrance barriers for mobility providers should be decreased by facilitating the integration of their service offerings into ticket systems (Nguyen et al., 2019) to improve competition in public transportation (Bundeskartellamt, 2019, 2022). The technical subsystem can support the achievement of those goals by facilitating the joint operation of an MaaS ticket system without dependencies on central organizations.

Decentralization of ticket systems can support the attainment of MaaS goals. From a social perspective, the inclusion of new mobility providers and their services is not moderated by a central organization (e.g., a single organization controlling the ticket system) that pursues economic interests like in several centralized ticket systems. Instead, the whole ticket federation needs to agree on changes to the federation. Thereby, the inclusion of new mobility providers and their services into the ticket system is less hindered by interests of central organizations. From a technical perspective, all mobility services in a decentralized MaaS ticket system become usable for customers with a single account, which can increase convenience and flexibility.

2.3 Using Distributed Ledger Technology in Decentralized Mobility-as-a-Service Ticket Systems

DLT can support the realization of social and technical benefits of decentralization for MaaS by enabling the decentralized operation of the core of MaaS ticket systems: a ledger. Each mobility provider can contribute to the operation of the MaaS ticket system with their own node, which stores and maintains a replication of the ledger. Through replication, availability of the ticket system can be increased. Moreover, cryptographic techniques and fault-tolerant consensus mechanisms can protect the integrity of the ledger. From a social perspective, mobility providers jointly contribute to the operation of the ticket system. In return, all mobility providers in a federation can be allowed to participate in decisions related to the ticket system and the ticket federation, decreasing dependencies on central organizations. However, incentives to use a decentralized MaaS ticket system may not apply to all mobility providers. For example, small mobility providers may be required to operate their own node in a federation, which produces costs for building up necessary competencies. Mobility providers need to directly compete with other providers, whose services have previously not been offered via the same ticket system. We assume the attainment of political MaaS goals and the corresponding alignment with societal norms, laws, and regulations to be a shared incentive of mobility providers for using decentralized MaaS ticket systems and that mobility providers prioritize this shared political incentive over individual motivations.

Regarding the design, implementation, and operation of decentralized MaaS ticket systems, we identify three core challenges, which relate to *(1) development, operation, and maintenance, (2) confidentiality*, and *(3) scalability*. We describe these challenges in the following.

Development, Operation, and Maintenance. In DLT-based ticket systems, all mobility providers need to agree on using exactly one DLT protocol which is executed on the nodes of all mobility providers. The DLT protocol constitutes the replicated state machine in the system. The DLT system must meet requirements of the ticket federation, such as achieving a minimum transaction processing rate (i.e., throughput), being scalable to a minimum number of peers, and preserving data confidentiality (see Sec. 2.2). To meet these requirements, several

design decisions related to the DLT system must be made, for example, selecting a suitable consensus mechanism, defining appropriate logic for the state machine, and assigning decision rights and accountabilities to mobility providers. All mobility providers in a federation can participate in decisions, for example, if they take on responsibilities for the development, operation, or maintenance of the system. However, all mobility providers have individual, potentially contradicting incentives. Compromises need to be found in negotiations. With an increasing number of mobility providers that participate in negotiations, the enforcement of decisions (e.g., on DLT protocol updates; Kannengießer et al., 2020) can slow down. Ticket federations may even split if mobility providers cannot agree on decisions (De Filippi & Loveluck, 2016). To preserve successful collaboration in decentralized ticket federations, an appropriate governance model is required, including mechanisms that facilitate the coordination of ticket federations under consideration of individual incentives of mobility providers (Beck et al., 2018).

Confidentiality. Only authorized entities must be able to access data stored in the ticket system. For example, a ticket inspector must be able to verify that customers are allowed to use mobility services, while mobility providers must not be able to track customer travels based on proofs-of-interaction stored in the ledger. Because the ledger is replicated across all nodes in the DLT system, all mobility providers that host nodes can access the ledger, including travel-related transactions. In this work, a transaction refers to a data set representing a proof-of-interaction (i.e., check-in or check-out) that marks the beginning and the ending of a service usage in pay-as-you-go ticket systems. Although DLT systems replicate their ledger across all nodes in the system, using DLT in ticket systems must not lead to unauthorized disclosure of business secrets and customer data in ticket systems.

Scalability. To achieve consistency across all replications of the ledger, nodes in DLT systems are synchronized by a consensus mechanism. Decentralized consensus mechanisms often rely on voting-like processes, where each peer (i.e., a node that participates in consensus finding by validating and verifying transactions) individually decides on each state transition. Peers extensively communicate to reach consensus producing communication overhead, which often depends on the number of peers involved in consensus finding (Kannengießer et al., 2020): the more peers, the longer the time for consensus finding (Dinh et al., 2017). Because any mobility provider should be able to join the ticket federation with an own peer, while transaction processing time must still be appropriate, the trade-off between scalability in the number of peers and scalability in terms of throughput as well as confirmation latency must be resolved appropriately for MaaS ticket systems.

2.4 Design Options to Solve the Core Challenges of Using Distributed Ledger Technology in Mobility-as-a-Service Ticket Systems

DLT systems can be private or public. Public DLT systems allow anyone to join with an own node that downloads a replica of the ledger. Private DLT systems only allow specified nodes to join. Besides the decision on a private or public system, technical challenges in DLT systems can be tackled by two principal categories of design options: *off-ledger* and *on-ledger*.

Off-ledger. Off-ledger systems are operated external to a distributed ledger and extend functionalities of DLT systems, such as by integrating real-world data. A variant of off-ledger systems to improve confidentiality in DLT system is the encryption of transaction payloads prior to their issuance to a DLT system (Brousmiche et al., 2018; Friebe et al., 2018; Stengele et al., 2021). However, encrypting data using secrets that are only known to a small group of individuals or single individuals in the ticket system limits the flexibility of billing and customer support functions in MaaS ticket systems. For example, if mobility providers encrypt transaction data on behalf of their customers, mobility providers can only decrypt and read proofs-of-interaction that are encrypted with their own secrets. Fares that need to be received by the contract partner of a customer as well as revenues to be distributed

to other mobility providers cannot be calculated or verified by the customer and other mobility providers, which contradicts our requirements for the MaaS ticket system (see Sec. 2.1).

Another variant of off-ledger systems is represented by oracles (Heiss et al., 2019) that can be triggered by a DLT system, for example, to issue data to a DLT system. Using oracles can prevents the disclosure of confidential data to participants in DLT systems by keeping such data external to the system (Kannengießer et al., 2021). For example, customer data can be stored in off-ledger databases that are only accessible by the contracting partners of customers (i.e., mobility providers) and the customers themselves. Despite their benefits for confidentiality, using oracles can hinder the verifiability of correctness of data issued to DLT systems.

A special type of off-ledger systems is represented by payment channel networks, such as the Lightning network for the Bitcoin system (Poon & Dryja, 2016). Payment channel networks are implemented to increase the transaction throughput of DLT systems by processing transactions outside of the respective DLT systems between at least two entities (i.e., a channel). The final cumulated outcomes are enforced when an entity issues the channel balance in the form of a cumulated transaction to the DLT system. Although payment channel networks can improve throughput and decrease transaction cost, they require the operation of a system additional to a DLT system. Thus, using payment channel networks can increase the required efforts for design and operation of DLT systems.

On-ledger. On-ledger design options refer to approaches that aim to solve challenges of using DLT by modifications of the state machine protocol. To improve confidentiality, DLT protocols, such as Monero and Zcash, implement mechanisms to obfuscate transaction histories by cryptographic means. Additionally, there are solutions that replace the replication-based approach in DLT systems by secure multi-party computation with multiple data providers (e.g., Fetzer et al., 2022). However, in such cryptographic techniques, confidentiality goals are usually prioritized over high availability and system performance.

Another option is to integrate Trusted Execution Environments (TEEs) into on-ledger protocols. TEEs are computing environments set up as enclaves in CPUs (e.g., Intel SGX; Costan & Devadas, 2016) or as co-processors (e.g., Trusted Platform Module; Arthur et al., 2015) and are designed to ensure authenticity and integrity of algorithms and their execution as well as confidentiality of runtime states (e.g., input and output data, memory). By moving the trust anchor into a dedicated hardware component, TEEs allow approaching requirements for MaaS ticket systems for high confidentiality, strong integrity, high throughput, and scalability in the number of peers (Cheng et al., 2019; Liu et al., 2019). Using TEEs, replications of the ledger as well as the state machine logic are stored in an encrypted way so that only the TEE can read the ledger. Moreover, TEEs allow for the verification of data and code integrity through “attestation” so that malicious manipulations can be detected. As a result, adversarial capabilities are limited and TEEs allow the implementation of flexible and rule-based access and identity management. TEEs can improve performance of consensus mechanisms by eliminating the possibility for equivocation, thus decreasing synchronization overhead (Clement et al., 2012; Veronese et al., 2011). However, if the private key of a TEE is leaked, all encrypted data can be decrypted and disclosed. If a TEE is broken and the private key is lost, already encrypted data may never be decrypted, which can stop the appropriate operation of information systems. Functionalities of the ticket system that require the data to be decrypted cannot be fulfilled anymore. Moreover, entities depend on TEE providers that may not support their technologies over the complete time their TEEs are used (e.g., Intel deprecating SGX on consumer hardware; Toulas, 2022).

Conclusion. For our prototypical MaaS ticket system, we use a private DLT system based on TEEs. Private DLT systems do usually not allow unauthorized entities to download a replication of the ledger, thus achieving better confidentiality compared to public systems. Moreover, private DLT systems tend to offer better maintainability compared to public ones. We deem TEEs as especially suitable for our MaaS ticket system because of their

confidentiality guarantees and their potential to increase throughput and decrease confirmation latency of private DLT systems. Moreover, TEEs allow for the flexible implementation of DLT protocols and business logic.

3 A Prototypical Decentralized Mobility-as-a-Service Ticket System

3.1 Design of the Prototypical Mobility-as-a-Service Ticket System based on Distributed Ledger Technology and Trusted Execution Environment

Architecture. The architecture of our MaaS ticket system (see Fig. 2) corresponds to a DLT system and comprises three principal layers: *entry point layer*, *ordering service layer*, and *RSM layer*. Each mobility provider operates a local instance of each layer. All entities (e.g., customers, employees of mobility providers, and vehicles), which interact with the ticket system (e.g., via smartphones), can establish connections with the entry points of all mobility providers.

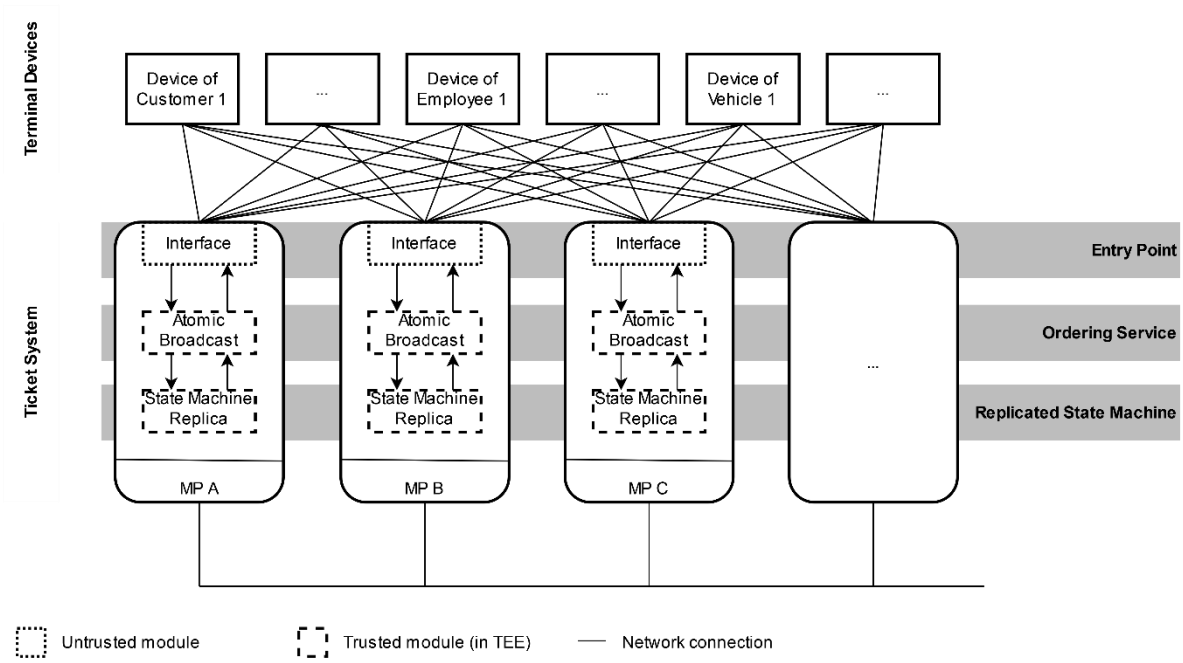

Fig. 2: System Architecture. Customers, employees, and vehicles connect to nodes of the decentralized TEE-based ticket system hosted by mobility providers (MP). Honest nodes will pass requests to the ordering service inside the TEE. Nodes communicate to agree on an order for a request. After the request is ordered, it is provided as input to the local state machine that applies the request to current system state following the business logic. The answer is then provided to the customer.

The core of our prototype is represented by an RSM and an ordering service. The RSM stores the business logic of the ticket system, including the ledger. The business logic specifies conditions and processes to control access to the ledger and manipulations of data stored in the ledger. To enable the state machine replicas of all mobility providers to deterministically derive identical results, they must operate on identical versions of the ledger. Thus, the ledger needs to be consistent across all nodes. To keep the ledgers consistent across nodes, the ordering service brings transactions (i.e., check-ins and check-outs) to be stored in the ledger in total order. The ordering service builds on a TEE-based Byzantine fault tolerant atomic broadcast middleware. Atomic broadcast is a consensus primitive that decides on the total order of transactions across all honest nodes in a distributed system. A Byzantine fault is the arbitrary deviation of a mobility provider from the DLT protocol, such as not being available or dispersing contradicting information to other mobility providers. Because we use TEEs to limit adversarial capabilities, the DLT protocol can be simplified and Byzantine behavior of up to 49% of the mobility providers participating in the DLT system can be tolerated while fulfilling availability requirements (Clement et al., 2012; Veronese

et al., 2011). Non-faulty mobility providers cannot miss customer requests or get otherwise in an inconsistent state. After the atomic broadcast has decided on the transaction order, the request is passed to the RSM. Based on the transaction order, the RSM maintains a relational database that represents the ticket system state.

Confidential Proofs-of-Interaction. Customers keep two different kinds of identifiers (UIDs) that are unique in the federation: Customer UIDs and Trip UIDs. Both types of UIDs are generated randomly by customers themselves. The RSM rejects chosen IDs if they are not unique. Using Customer UIDs, the system can identify and authenticate customers. Customer UIDs are used for check-in and check-out operations and are exchanged between customers and the ticket system via encrypted communication channels.

If customers would use their Customer UIDs to interact with vehicles or ticket inspectors, the customer behavior would be observable outside the TEE-based ticket system. Thus, customers use their Trip UIDs as pseudonyms when identifying themselves in public, i.e., when interacting with infrastructure (e.g., vehicles) and ticket inspectors. Because Trip UIDs are invalidated and newly generated after every check-in and check-out so that individual trips cannot be linked, Trip UIDs can be exchanged via unencrypted communication channels. If the generation of UIDs is truly random, Trip UIDs can only be linked within the TEE by the RSM because the Trip UID history of a customer is only available inside the TEEs in the ticket system. The TEEs only return aggregates of interaction history meta data (e.g., fares based on cumulated travel time or distances).

To check in (or check out), customers transmit their current Trip UID to the vehicle they want to enter (or leave). The vehicle cryptographically signs the Trip UID and meta data, including the current location and its Vehicle UID, and transmits the information back to the customer. The customer sends their Customer UID and the data received from the vehicle to the ticket system to store the check-in. Ticket inspectors of any mobility provider can verify whether customers are allowed to use mobility services. Ticket inspectors send the Trip UID disclosed by the customer and the Vehicle UID to the ticket system. The ticket system now verifies whether the Trip UID is linked to a Customer UID that is checked-in with the vehicle and returns a binary result to the ticket inspector.

Revenue Distribution. In a pay-as-you-go tariff, the proofs-of-interaction are used to calculate tailored fares within agreed billing intervals (see Sec. 2.1). Whenever a billing interval ends, the RSM collects all check-ins and check-outs logged in the interval to calculate fares to be paid by customers. As the RSM operates inside the TEE, the RSM can access all proofs-of-interaction in plain text and can reconstruct proofs-of-interaction histories of each customer. Upon requests by mobility providers to execute the billing function, the RSM returns the billing results and the revenue distribution. However, to protect the customers' privacy and the mobility providers' business secrets (e.g., service utilization), each mobility provider only learns the fares to be collected from their contract partners and what payments to other federation members must be made to correctly distribute revenues.

3.2 Preliminary Evaluation

Although TEEs are promising to solve technical challenges in decentralized MaaS ticket system, it is unclear if a TEE-based atomic broadcast is a suitable ordering service layer in real-world scenarios. To evaluate whether TEE-based atomic broadcast can achieve the needed performance, we implement MinBFT (Veronese et al., 2011). We chose MinBFT as a representative for TEE-based atomic broadcasts because almost all recent approaches are based on ideas of MinBFT and MinBFT represents a baseline for the performance of TEE-based atomic broadcast. We investigate the confirmation latency and throughput of MinBFT for a workload model derived from the performance indicators of the "Verkehrsverbund Berlin-Brandenburg (VBB)", one of the largest transport associations in Europe. We map the performance indicators of the VBB to our decentralized MaaS ticket system. In 2019, the VBB was comprised of 36 mobility providers, whose mobility services were used by 4.3 million customers per day

(Verkehrsverbund Berlin-Brandenburg, 2021b, 2021a). Assuming that 50% of the daily customers used the services in a three-hour morning rush hour, there are 2.15 million check-in and 2.15 million check-out operations in the rush hour. Because every check-in and check-out corresponds to a request to the MaaS ticket system, the ticket system, including the ordering service (i.e., MinBFT), must handle about 400 requests per second when all check-ins and check-outs are equally distributed over the three-hour time window.

We implemented a benchmarking framework and toolkit for RSM-based systems in the Rust programming language. We used Intel's Software Guard Extensions (SGX; Costan & Devadas, 2016) to setup TEEs and the Apache Teaclave SGX SDK (https://github.com/apache/incubator-teaclave-sgx-sdk) to enable the use of Rust inside SGX enclaves. We deployed our implementation of MinBFT in our benchmarking framework to ten equal servers with Ubuntu 20.04.1, Linux kernel 5.13.0, Intel Xeon E-2288G CPU, and 32GB main memory. A separate server was used to coordinate the experiment and generate client requests. All eleven servers are placed in the same data center with a 10 Gbit network. The framework generates a configurable number of requests that are sent to all nodes every second on average. We approximate the encrypted requests of our proposed protocol by generating 512 Bytes of random data for every request. 512 Bytes are an upper estimate for the data to be exchanged at check-in and check-out respectively. For every request, the confirmation latency is stored. To increase the throughput and decrease confirmation latency, MinBFT batches requests. A batch has a maximum size of 200 requests. We conducted the experiment for 20, 40, and 60 mobility providers with a workload of 100 up to 1,000 parallel requests per second. Each configuration was executed 30 times for 150 seconds. To observe only a stable system, the first 20 seconds and last 10 seconds of each run were not measured.

Our results are depicted in Fig. 3. More than twice the required number of parallel requests by the VBB model can be handled by MinBFT within 500 milliseconds for a ticket system of 60 mobility providers. Adding the usual network latencies (e.g., between datacenters and for reaching the ticket system via mobile networks) results in response times of a few seconds which we consider realistic and fast enough. The observed confirmation latencies for 20 mobility providers are, in contrast to our expectations, not monotonic. We suspect the generated load being too low to optimally utilize the batch size of 200 requests resulting in non-monotonic latency patterns.

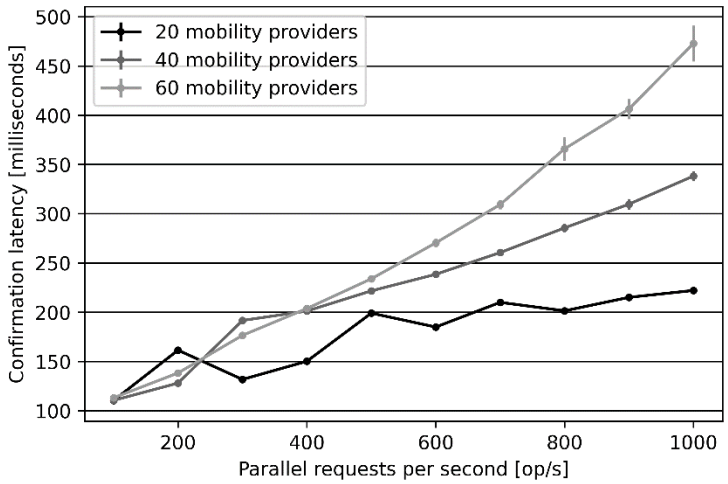

Fig. 3: Confirmation latency for 100 to 1,000 parallel requests per second and 20 to 60 mobility providers. The error bars indicate the 95% confidence interval. Even for 60 mobility providers, MinBFT handles up to 1,000 requests per second within 500 milliseconds.

4 Discussion

To support the fulfillment of MaaS goals, we present the design of a prototypical MaaS ticket system that can be operated in a decentralized way. In the design, we considered key requirements related to confidentiality of proofs-of-interaction, strong integrity of data stored in the ledger, high availability, and fast transaction processing. To understand the extent to which our system can meet those requirements, we implemented the ordering service of the system based on the TEE-based Byzantine fault tolerant atomic broadcast protocol MinBFT and evaluated the system in a benchmark. Our preliminary evaluation indicates the technical feasibility of decentralized MaaS ticket systems based on the RSM concept and TEEs.

In our benchmark, we observed a strong impact of the size of request batches on the confirmation latency and throughput in the RSM. Large batches (e.g., at least 1,000 requests per batch) can slow down the system, while small batches (e.g., up to 200 requests per batch) can accelerate transaction processing in terms of confirmation latency and throughput. However, choosing too small batch sizes in rather large federations (i.e., more than 40 mobility providers) with request rates of more than 200 requests per second may lead to system overloading. Additionally, our benchmark shows non-monotonic behavior of the MaaS ticket system (see Fig. 3) for small federation sizes. We suspect such behavior to also be related to the choice of batch size, request rates, and the way our framework generates client requests. If the number of requests generated in an interval does not fill up a batch, some requests may need to wait for the next batch resulting in higher confirmation latencies.

This work supports the development of decentralized MaaS ticket systems by informing about common challenges related to using RSMs, presenting a prototypical system design of a decentralized MaaS ticket system that can cope with those challenges, and describing initial evaluation results regarding performance characteristics of the system. The design of our prototypical MaaS ticket system can serve as a starting point for the development and instantiation of fully-fledged decentralized ticket systems to reach MaaS goals. Our benchmarking framework for decentralized TEE-based RSMs represents an initial approach for the investigation of behaviors of decentralized MaaS ticket systems. In contrast to previous approaches in literature, our benchmarking framework aims at generating realistic workload models. Through such investigations, the technical suitability of TEE-based decentralized MaaS ticket systems for real-world applications can be better understood, for example, in terms of throughput and transaction processing time.

For the design of the MaaS ticket system, we made several assumptions for the social subsystem and the political environment that reduce the actual complexity of decentralized information systems. For example, we assumed that each mobility provider is willing to participate in a ticket federation, to potentially give up market power, and to take on additional tasks (e.g., operation of an own node). Thus, our evaluation only allows to draw conclusions about the technical feasibility of the ticket system, while social and political impacts on the ticket system need to be investigated to assess the practicality of our prototype in real-world scenarios. Moreover, we largely derived requirements for MaaS ticket systems from literature (e.g., Nguyen et al., 2019; Preece & Easton, 2019; Verkehrsverbund Berlin-Brandenburg, 2021a). In future research, these requirements need to be advanced in collaboration with mobility providers to improve authenticity of the investigated MaaS scenario. We needed to simplify certain technical aspects in our evaluation. For example, we did not investigate potential effects of network latencies between clients and the ticket system or if nodes are distributed across different data centers, which increase confirmation latency (Veronese et al., 2011). We are planning to implement latency emulation to improve the external validity of our benchmarking model.

Our preliminary evaluation indicates the relevance of atomic broadcast parameterization. Research on rate limiting to prevent system overload and on load-adaptive batching seems to be relevant to implement DLT-based MaaS ticket systems for real-world scenarios. Additionally, current research has identified issues of atomic broadcast algorithms with node crashes (Spiegelman et al., 2022). Because of the requirements of high availability and maintainability, we consider investigations of the performance and viability of the ticket system in cases of node maintenance, node crash and node recovery as very important for the MaaS scenario. Moreover, our benchmarks are focused on the behavior of the ordering service. To investigate the behavior of the MaaS ticket system more thoroughly, our framework needs to be extended to also include the overall RSM in measurements.

References

Arthur, W., Challener, D., & Goldman, K. (2015). *A Practical Guide to TPM 2.0* (1st ed.). Apress Berkeley, CA, USA.

Beck, R., Müller-Bloch, C., & King, J. L. (2018). Governance in the Blockchain Economy: A Framework and Research Agenda. *Journal of the Association for Information Systems*, *19*, 1020–1034.

Brousmiche, L., Durand, A., Heno, T., Poulain, C., Dalmieres, A., & Ben Hamida, E. (2018). Hybrid Cryptographic Protocol for Secure Vehicle Data Sharing Over a Consortium Blockchain. *2018 IEEE Int. Conf. on Internet of Things and IEEE Green Computing and Communications and IEEE Cyber, Physical and Social Computing and IEEE Smart Data*, 1281–1286.

Bundeskartellamt. (2016, May 24). *Deutsche Bahn AG to make changes to ticket sales*. https://www.bundeskartellamt.de/SharedDocs/Meldung/EN/Pressemitteilungen/2016/24_05_2016_DB_Fahrkarten.html;jsessionid=5A33E20004FB802D81BC637A5C2E14E0.1_cid390?nn=3591568

Bundeskartellamt. (2019, November 28). *Proceeding against Deutsche Bahn AG - Bundeskartellamt examines possible anticompetitive impediment of mobility platforms*. https://www.bundeskartellamt.de/SharedDocs/Meldung/EN/Pressemitteilungen/2019/28_11_2019_DB_Mobilitaet.html;jsessionid=5B080CC68C5A18D92570B7F8B06B572D.1_cid390?nn=3591568

Bundeskartellamt. (2022, April 20). *Fairer Wettbewerb um digitale Mobilitätsdienstleistungen – Bundeskartellamt mahnt Deutsche Bahn wegen möglicher Behinderung von Mobilitätsplattformen ab*. https://www.bundeskartellamt.de/SharedDocs/Meldung/DE/Pressemitteilungen/2022/20_04_2022_Bahn.html

Castro, M., & Liskov, B. (1999). Practical Byzantine Fault Tolerance. *Proc. Third Symposium on Operating Systems Design and Implementation*, 173–186.

Chatterjee, S., Sarker, S., Lee, M. J., Xiao, X., & Elbanna, A. (2021). A possible conceptualization of the information systems artifact: A general systems theory perspective. *Information Systems Journal*, *31*(4), 550–578.

Cheng, R., Zhang, F., Kos, J., He, W., Hynes, N., Johnson, N., Juels, A., Miller, A., & Song, D. (2019). Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contracts. *2019 IEEE European Symposium on Security and Privacy*, 185–200.

Clement, A., Junqueira, F., Kate, A., & Rodrigues, R. (2012). On the (limited) power of non-equivocation. *Proc. 2012 ACM Symposium on Principles of Distributed Computing*, 301–308.

Costan, V., & Devadas, S. (2016). Intel SGX Explained. *Cryptology EPrint Archive*.

De Filippi, P., & Loveluck, B. (2016). The invisible politics of Bitcoin: Governance crisis of a decentralised infrastructure. *Internet Policy Review*, *5*(3).

Dinh, T. T. A., Wang, J., Chen, G., Liu, R., Ooi, B. C., & Tan, K.-L. (2017). BLOCKBENCH: A Framework for Analyzing Private Blockchains. *Proc. 2017 ACM Int. Conf. on Management of Data*, 1085–1100.

European Environment Agency. (2021). *Greenhouse gas emissions from transport in Europe*.

Farsi, M., Fetz, A., & Filippini, M. (2007). Economies of Scale and Scope in Local Public Transportation. *Journal of Transport Economics and Policy*, *41*(3), 345–361.

Fetzer, V., Keller, M., Maier, S., Raiber, M., Rupp, A., & Schwerdt, R. (2022). PUBA: Privacy-Preserving User-Data Bookkeeping and Analytics. *Proc. Privacy Enhancing Technologies*, *2022*(2), 447–516.

Friebe, S., Sobik, I., & Zitterbart, M. (2018). DecentID: Decentralized and Privacy-Preserving Identity Storage System Using Smart Contracts. *2018 17th IEEE Int. Conf. On Trust, Security And Privacy In Computing And Communications/ 12th IEEE Int. Conf. On Big Data Science And Engineering*, 37–42.

Gould, E., Wehrmeyer, W., & Leach, M. (2015). *Transition pathways of e-mobility services*. 349–359.

Gupta, D. (2016). *Towards Performance and Dependability Benchmarking of Distributed Fault Tolerance Protocols* [PhD Thesis, Université Grenoble Alpes].

Heines, R., Kannengießer, N., Sturm, B., & Sunyaev, A. (2021). Need for Change: Business Functions Affected by the Use of Decentralized Information Systems. *International Conference on Information Systems 2021 Proceedings*. International Conference on Information Systems 2021, Austin, Texas, USA.

Heiss, J., Eberhardt, J., & Tai, S. (2019). From Oracles to Trustworthy Data On-Chaining Systems. *2019 IEEE Int. Conf. on Blockchain*, 496–503.

Jittrapirom, P., Caiati, V., Feneri, A.-M., Ebrahimigharehbaghi, S., González, M. J. A., & Narayan, J. (2017). Mobility as a Service: A Critical Review of Definitions, Assessments of Schemes, and Key Challenges. *Urban Planning*, *2*(2), 13–25.

Kannengießer, N., Lins, S., Dehling, T., & Sunyaev, A. (2020). Trade-Offs between Distributed Ledger Technology Characteristics. *ACM Computing Surveys*, *53*(2).

Kannengießer, N., Lins, S., Sander, C., Winter, K., Frey, H., & Sunyaev, A. (2021). Challenges and Common Solutions in Smart Contract Development. *IEEE Transactions on Software Engineering*. IEEE Transactions on Software Engineering.

Lamberti, R., Fries, C., Lücking, M., Manke, R., Kannengießer, N., Sturm, B., Komarov, M. M., Stork, W., & Sunyaev, A. (2019). An Open Multimodal Mobility Platform Based on Distributed Ledger Technology. In O. Galinina, S. Andreev, S. Balandin, & Y. Koucheryavy (Eds.), *Internet of Things, Smart Spaces, and Next Generation Networks and Systems* (pp. 41–52). Springer International Publishing.

Li, W., Meese, C., Guo, H., & Nejad, M. (2020). Blockchain-enabled Identity Verification for Safe Ridesharing Leveraging Zero-Knowledge Proof. *ArXiv:2010.14037 [Cs]*.

Liu, J., Li, W., Karame, G. O., & Asokan, N. (2019). Scalable Byzantine Consensus via Hardware-Assisted Secret Sharing. *IEEE Transactions on Computers*, *68*(1), 139–151.

Nguyen, T. H., Partala, J., & Pirttikangas, S. (2019). Blockchain-Based Mobility-as-a-Service. *2019 28th Int. Conf. on Computer Communication and Networks*, 1–6.

Pease, M., Shostak, R., & Lamport, L. (1980). Reaching Agreement in the Presence of Faults. *Journal of the ACM*, *27*(2), 228–234.

Poon, J., & Dryja, T. (2016). *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*.

Preece, J. D., & Easton, J. M. (2019). Blockchain Technology as a Mechanism for Digital Railway Ticketing. *2019 IEEE Int. Conf. on Big Data*, 3599–3606.

Schneider, F. B. (1990). Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys*, *22*(4), 299–319.

Spiegelman, A., Giridharan, N., Sonnino, A., & Kokoris-Kogias, L. (2022). *Bullshark: DAG BFT Protocols Made Practical* (arXiv:2201.05677). arXiv.

Stengele, O., Raiber, M., Müller-Quade, J., & Hartenstein, H. (2021). ETHTID: Deployable Threshold Information Disclosure on Ethereum. *2021 Third Int. Conf. on Blockchain Computing and Applications*, 127–134.

Toulas, B. (2022, January 14). *New Intel chips won't play Blu-ray disks due to SGX deprecation*. BleepingComputer. https://www.bleepingcomputer.com/news/security/new-intel-chips-wont-play-blu-ray-disks-due-to-sgx-deprecation/

Verkehrsverbund Berlin-Brandenburg. (2021a). *Die wichtigsten Kennzahlen im Verkehrsverbund Berlin-Brandenburg*. https://www.vbb.de/fileadmin/user_upload/VBB/Dokumente/Verkehrsverbund/60_wichtigste_VBB-Kennzahlen_2021_01.pdf

Verkehrsverbund Berlin-Brandenburg. (2021b). *Zahlen & Fakten 2021*. https://www.vbb.de/fileadmin/user_upload/VBB/Dokumente/Verkehrsverbund/zdf-2021.pdf

Veronese, G. S., Correia, M., Bessani, A. N., Lung, L. C., & Verissimo, P. (2011). Efficient Byzantine Fault-Tolerance. *IEEE Transactions on Computers*, *62*(1), 16–30.

Wang, D., & Zhang, X. (2021). Secure Ride-Sharing Services Based on a Consortium Blockchain. *IEEE Internet of Things Journal*, *8*(4), 2976–2991.