velink - A Blockchain-based Shared Mobility Platform for Private and Commercial Vehicles utilizing ERC-721 Tokens

Dominic Pirker*†, Thomas Fischer*†, Harald Witschnig†, Christian Steger*
Email: {dominic.pirker, thomas.fischer3, harald.witschnig}@infineon.com, steger@tugraz.at
*Institute for Technical Informatics, Graz University of Technology, Graz, Austria
†Development Center Graz, Infineon Technologies AG, Graz, Austria

***Abstract*—Transportation of people and goods is important and crucial in the context of smart cities. The trend in regard of people's mobility is moving from privately owned vehicles towards shared mobility. This trend is even stronger in urban areas, where space for parking is limited, and the mobility is supported by the public transport system, which lowers the need for private vehicles. Several challenges and barriers of currently available solutions retard a massive growth of this mobility option, such as the trust problem, data monopolism, or intermediary costs.**

Decentralizing mobility management is a promising approach to solve the current problems of the mobility market, allowing to move towards a more usable internet of mobility and smart transportation. Leveraging blockchain technology allows to cut intermediary costs, by utilizing smart contracts. Important in this ecosystem is the proof of identity of participants in the blockchain network. To proof the possession of the claimed identity, the private key corresponding to the wallet address is utilized, and therefore essential to protect. In this paper, a blockchain-based shared mobility platform is proposed and a proof-of-concept is shown. First, current problems and state-of-the-art systems are analyzed. Then, a decentralized concept is built based on ERC-721 tokens, implemented in a smart contract, and augmented with a Hardware Security Module (HSM) to protect the confidential key material. Finally, the system is evaluated and compared against state-of-the-art solutions.

***Index Terms*—shared mobility, blockchain, smart contract, Ethereum, token, ERC-721, HSM**

I. Introduction

Shared mobility services are expanding over the last years, especially in urban areas, where less people own private vehicles. Goal of this rapidly growing concept is to make it convenient for consumers to access vehicles of any kind, whenever they need. Besides avoiding the expenses for buying a vehicle, other aspects in regard of cost-saving are insurance, parking, and many more. Another problem solved with this concept, at least partly, is the problem of scare space, due to the high quantity of parked vehicles in the city, since privately owned vehicles are less in use- than in idle-mode.

Unfortunately, the concept of shared mobility as it is designed now, not only brings advantages. Many providers as well as new modes of transport are pushing into the market. This leads to an uncontrolled growths of available transportation offers. Consumer of these services are having difficulties to get best out of this ecosystem, since many different service providers - each with their own application - are available to choose, as highlighted in Fig. 1. Not only the time consuming registration process for every provider, but also the non-existent overview of prices impede people from using these services.

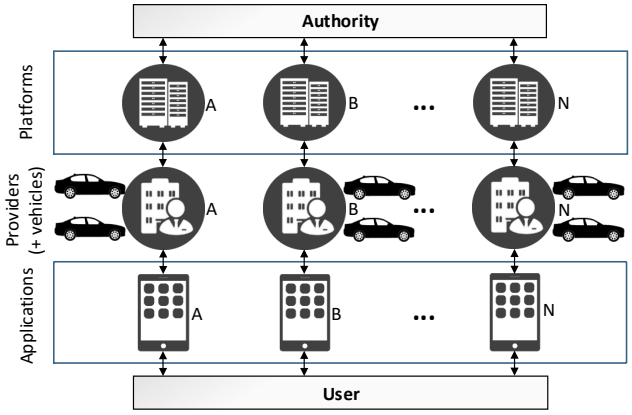

Fig. 1. Current ecosystem with distinct applications and platforms for each shared mobility provider (A, B, ... N)

Furthermore, the rapid growth of new mobility services within cities raises concerns regarding their huge impact on the cities infrastructure. Thus, big providers are getting more and more powerful, since they have all the data to comprehend the impact on traffic, people's behavior, and their new habits. Based on that, they can foster their enlargement and reach monopolism, which shall be avoided in this and any other area. Central and hierarchical concepts promote the growth of monopolism, which lead to the decision to design the concept proposed in this work in a decentralized manner.

Third problem for consumers of the shared mobility market is that intermediaries need to be paid for acting as a middleman. This could be avoided by designing the architectures based on the concept of decentralization.

Within this work, renting and sharing is considered equivalent, since for both there need to be an owner and a tenant.

The main contributions of this work are:

- Design of a secured and decentralized architecture for a shared mobility platform, enabled by extending vehicles with an HSM to protect important key material.

This is the full version of a paper which appears in 2021 IEEE 5th International Conference on Cryptography, Security and Privacy (CSP).
The proceedings version is available online at: https://doi.org/10.1109/CSP51677.2021.9357605

- Implementation of a proof-of-concept.
- Evaluation of the designed system, including comparison to available state-of-the-art solutions.

II. Background

Blockchain is an emerging technology which started in 2008 with the cryptocurrency Bitcoin. Nowadays the decentralized network concept is utilized in a broad range of applications. This is mainly enabled with smart contracts. A smart contract is a computer program running as part of the network protocol, such as Ethereum. A description of smart contracts is given in [1]. With smart contracts, tokenization of assets is enabled. Additionally, required interaction for the given application is defined. The most used standards for Ethereum tokens are the *ERC-20 Token Standard* and the *ERC-721 Token Standard* [2]. In order to create these tokens, they have to be minted, as physical coins. Therefore, smart contracts need to define a function to mint these tokens. Within this work, the focus is on the ERC-721 tokens, since they are utilized in the proposed concept. They are non-fungible and mostly represent unique digital entities. ERC-721 is a free and open standard to build non-fungible tokens on the Ethereum blockchain [3]. The ERC-721 standard defines a minimum interface to allow unique tokens to be managed, owned, and traded [3]. Most prominent example for ERC-721 tokens are *CryptoKitties*, which are gamified collectables [2].

III. Related Work

The shared mobility market is already dominated by big players in various areas, whose system design is based on a centralized architecture. In Germany, *ShareNow* is the dominating company, with 4.2 million registered users by 2019 [4]. In [5], security related threats, which are mostly present in centralized systems are analyzed. Examples for those threats are [5]: unauthorized access; Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks; sensor related issues; data transfer issues; IoT web security; information sharing and storage.

Many of them are solvable by leveraging decentralized architectures, if implemented and designed properly. In [5], important features consolidating the role of blockchain in Internet of Vehicles (IoV) are listed. Examples are immutability, decentralization, consensus, security, transparency, and many more. A detailed description and their effect are elaborated in [5]. Due to these features, solving many current threats, this work is focusing on decentralized architectures, hence *ShareNow* is not analyzed. In the remaining section, state-of-the-art systems and concepts are summarized.

In [6], five concept are analyzed, where the most promising concepts are *HireGo* and *DAV*.

- The first one is based on Ethereum smart contracts, with the design disadvantage, that the vehicle token is transferred to the tenant, which leads to an ownership loss during the time where the rental is active.
- The latter concept is not only focusing on cars, but also on trucks, drones, and more to build a network of ultimately connected self-driving vehicles [7].

Besides the concepts analyzed in [6], four additional concept are considered as state-of-the-art for the given problem statement within the shared mobility ecosystem. In [4], two different concepts for a privacy-preserving blockchain-based system for car sharing are described, where both leveraging zero-knowledge protocols. Several parties, such as authorities and industry, are considered to be part of the ecosystem.

- The first solution is based on zkSNARKS, a zero-knowledge protocol utilized at the Ethereum platform. As a toolbox for zkSNARKS, ZoKrates was used [4]. The concept consists of three phases: initialization phase, attribute issuance, and attribute disclosure. The first phase is responsible for environment setup, key generation, contract deployment, and issuer registration for industry and authorities. In the second phase, credentials for user data are issued. The last phase is for finally using the service and has two main objectives. First, the user is revised for being the real owner, and second, attributes such as validity of driving license are verified [4].
- Second concept described in [4] is leveraging the Hyperledger Indy framework. The proposed system setup is designed to implement two distinct blockchain networks for authorities and industry respectively. Each party can derive attributes to be proven by the verifier for consuming a service, such as renting a car [4]. The user has to request membership credentials for both blockchain network, and store them in their digital wallet. Then the proof details are read from the car, which is acting as a verifier. A composite proof is constructed accordingly and presented to the car. Then, access to the car is granted or denied respectively.
- The proposed concept in [6] introduces a blockchain-based car-sharing platform, built upon the Ethereum blockchain. It is based on two different token types, the non-fungible token type ERC-721 and the fungible token type ERC-20. The difference is explained in [2]. First token represent car assets, second represents unlock tokens, which are bought with ETH. Additionally, the defined token of the latter type is used to reward users when using this system. To reduce costs of Ethereum transactions, the car image is stored at InterPlanetary File System (IPFS). As a consequence, only the IPFS URI needs to be stored in the blockchain network [6].
- Another concept is *Cryptober*, a blockchain-based, secure, and cost-optimal car rental platform [8]. This concept requires car, owner, and tenant to be registered at the platform. When renting a car, the tenant checks the database of available cars and afterwards contacts the owner. In [8], the modality of communication is not defined. The involved parties are required to meet at a mutual decided location for checking the condition of the car and to define parameters such as maximum

speed and maximum distance. In case both agree with the conditions, a *start* block is generated containing the before defined rules. After the ride, both parties meet again and a *finish* block is generated. Besides the basic ride, emergency cases are also considered, but they are not focus for this work. The underlying blockchain network is not specified in the context of [8].

IV. Concept and Architecture

This chapter proposes *velink*, a blockchain-based shared mobility platform, running on Ethereum and utilizing ERC-721 tokens, for users and vehicles.

The following requirements for the system design have been identified:

- High level of security, by integrating an HSM for storing confidential key material and signing transactions.
- Simple participation on the platform for users and providers.
- Avoid monopolism in regard of data and market share.

A. Overview

The concept proposed in this work is based on an Ethereum smart contract. Fig. 2 depicts the system design. All parties involved in the car sharing procedure are interacting with the smart contract. The following sections describe the procedures required to happen prior and during renting a vehicle.

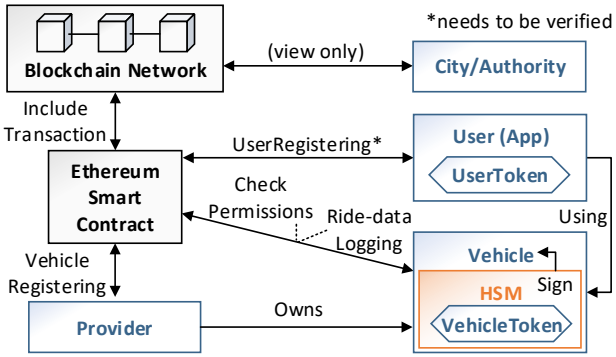

Fig. 2. Overview of architecture design of proposed shared mobility platform

B. Registration Procedure

User and vehicle need to be registered in the blockchain network in order to participate in the system. The provider needs to be trusted, but the procedure to ensure this is out of scope in this work. For private vehicles, the vehicle owner is considered as provider. All trusted providers register their vehicles by triggering their vehicles to call the function *MintVehicleToken(...)* of the smart contract. This function generates the vehicle token, as explained in the background section. The result is an ERC-721 token, hereafter referred to as vehicle token, owned by the vehicle itself. The vehicle is equipped with an HSM to store confidential key material, as depicted in Fig. 2. The registration procedure is depicted in Fig. 3. Before generating the vehicle token, the vehicle must have an address to be accessible via the blockchain network. Therefore the HSM is utilized to generate a key pair consisting of a public and a private key. The private key is randomly generated and used to sign transactions. The address of the vehicle is derived from the public key and hereafter referred to as wallet address. This wallet contains the vehicle token. To proof the possession of the vehicle (token), the private key is utilized. The private key never leaves the HSM, since a compromised private key leads to an ownership loss.

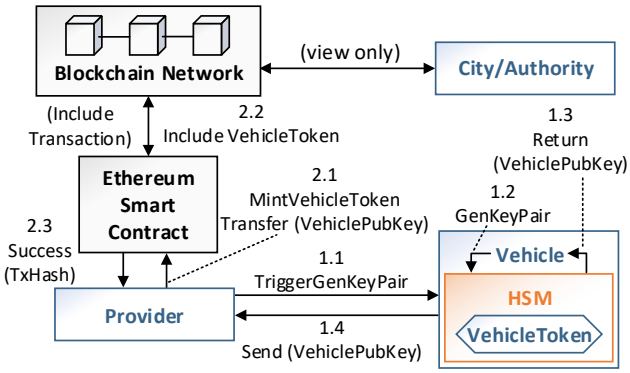

Fig. 3. Vehicle registration procedure triggered by provider

As stated in the beginning, the user (tenant) needs to be registered too, in order to get access to vehicles. The general procedure is equivalent to the vehicle registration, but the user data, such as validity of driving license, needs to be verified before a user token is minted. Therefore several options are possible. One option is to have an authority checking the details before registering. Another possibility is to reuse an existing account from a shared mobility provider and generate a user token based on this information. This approach is the most convenient for the end user, since registration and the validation of driving licenses is already done, and no additional interaction is required. For this approach, the provider triggers the *MintUserToken(...)* function of the smart contract and transfers the ownership to the user. Again, the provider needs to be trusted by the user. Ideally, an HSM for storing the confidential key material, is utilized in the smart phone application, as in the vehicle.

C. Unlocking Procedure

After successful registration of user and provider, the system is operational. The unlocking procedure is depicted in Fig. 4.

The user sends the wallet address, derived from the corresponding public key, via Bluetooth to the vehicle. The vehicle is responsible for checking the permissions of the user. The advantage of having this step performed by the vehicle instead of the smart phone application, is the lower vulnerability against attacks. The *CheckPermission(...)* function of the smart contract is triggered with wallet addresses of user and vehicle as input parameters. User parameters such as driving license or permitted vehicle types are checked, and the corresponding authorizations are returned to the vehicle. In case the user is authenticated as an authorized user, the transaction for vehicle

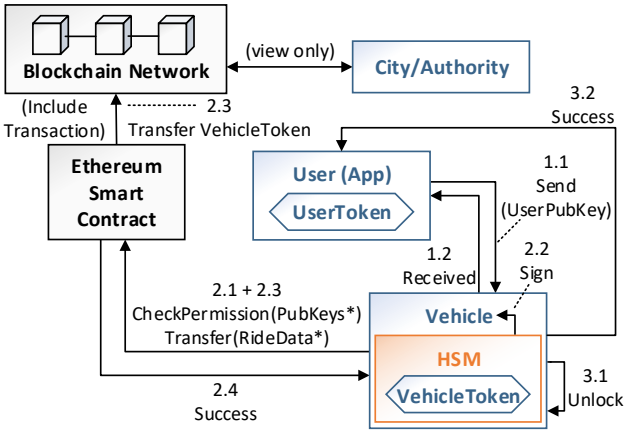

Fig. 4. Unlocking and permission validation procedure

unlocking is generated by triggering the *VehicleUnlock(...)* function of the smart contract. This results in sending the vehicle token to the vehicle itself, with modified parameters such as user, lock state, time stamp, or GPS position. With this measure, the vehicle never loses the ownership of the corresponding token. For generating the transaction signature, the securely stored private key of the vehicle, is utilized. Then, the transaction is transferred to the blockchain network and included in the next block, if the signature is valid. In case of success, the transaction hash is returned and the vehicle gets unlocked.

For locking the vehicle again, the procedure is equivalent, except two differences. First, the *LockVehicle(...)* function is called instead of the *UnlockVehicle(...)* function. Second, the vehicle token parameters are set respectively.

D. Smart Contract and Tokenization

In the proposed concept, the following non-fungible tokens, including their parameters are defined in the smart contract:

- *UserToken*: mintedBy, mintedAt, allowedVehicle
- *VehicleToken*: mintedBy, mintedAt, vehicleType, lockState, currentUser, position, etc.

Each token has a corresponding *MintToken(...)* function, defined in the smart contract. In addition to that and the interface defined in standard [3], following functions are implemented in the smart contract:

- *CheckPermission*: Used to validate the permission of the user to unlock and drive the corresponding vehicle.
- *VehicleUnlock*: Changes the lock state of the vehicle and sets the current user parameter.
- *VehicleLock*: Equivalent to *VehicleUnlock*, but sets the status to locked and clears current user.
- *GetVehicleLockStatus*: Return current status of corresponding vehicle.

The system functionality is partitioned between smart contract, host controller and HSM, as depicted in Table I.

TABLE I
PARTITIONING OF APPLICATION FUNCTIONALITY

Smart Contract	Host Controller	HSM
MintVehicleToken	DeriveWalletAddress	GenerateKeyPair
MintUserToken	BuildTransaction	GetPublicKey
TransferToken	GetSignedTransaction	GenerateSignature
GetVehicleLockStatus	SendTransaction	
CheckPermission	GetPermission	
VehicleUnlockRaw	VehicleUnlock	
VehicleLockRaw	VehicleLock	

E. Privacy Enhancement

A key challenge of utilizing the blockchain technology for enterprise applications is data confidentiality, as stated in [9]. Several privacy preserving methods based on blockchain are already existent and explained in [4]. All explained schema are leveraging zero knowledge proofs, such as zkSNARKS, for a secured exchange of tokenized assets without a trusted third party. In [9], weaknesses and challenges of existing models are analyzed. The main challenges of these systems are the initial setup phase, which is prone to manipulation, and the lack of flexibility, since the application for many real world use cases is restricted [9]. This means, a zero knowledge proof such as zkSNARKS needs to be implemented and adopted properly, in order to successfully protect the privacy of participating parties. In [9], a novel schema for zero knowledge proof, solving weaknesses of state-of-the-art schema, is introduced. In this concept, privacy enhancement is not addressed but the chosen architecture allows the integration.

V. PROOF-OF-CONCEPT

To show the feasibility, a proof-of-concept was implemented. The demonstrator and the test transaction performed in the *Ropsten Testnet* are depicted in Fig. 5. It is built around an HSM from Infineon Technologies AG, which is visible in Fig. 5 in the car's trunk.

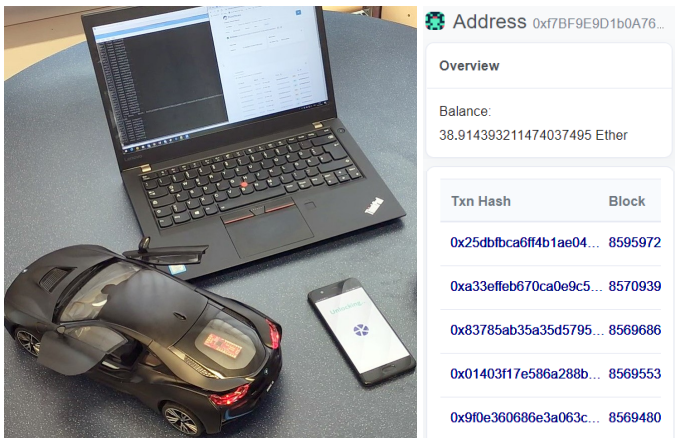

Fig. 5. Proof-of-concept with implemented HSM and test transactions performed in *Ropsten Testnet* (https://ropsten.etherscan.io/)

The demonstrator shows the unlocking and locking procedure of a shared car, based on the Ethereum blockchain, as depicted in Fig. 4. The registration of the user and the

vehicle is considered as a precondition, which has to happen before. Those steps were performed with Ethereum Remix, an interface for Ethereum-based blockchains. As described in Section IV-B, the registration of user and vehicle results in a respective token creation.

A. Components

The demonstrator mainly consists of three parts:

- *Smart phone*: Used to scan via Bluetooth for vehicles in the vicinity of the user and to trigger the (un)locking procedure as depicted in Fig. 6.
- *Car*: Represented by a toy car, equipped with a Raspberry Pi Zero W and an HSM. The Raspberry Pi is communicating with the utilized blockchain network.
- *Blockchain and smart contract*: Ethereum blockchain and smart contract defining the functionality as described in Section IV-D.

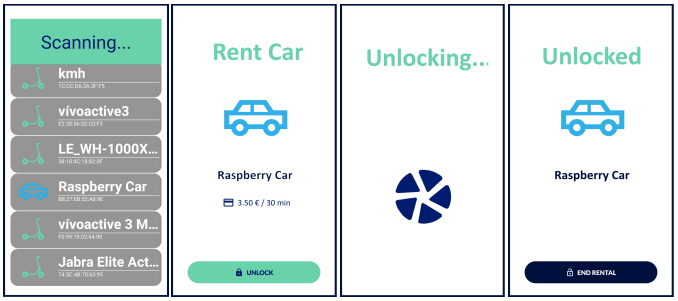

Fig. 6. Proof-of-concept end user smart phone application for scanning and unlocking vehicles with steps: scan, select, unlock, and success

B. Architecture

The demonstrator architecture is depicted in Fig. 7. For the smart phone, apps for Android and iOS were developed, to communicate with the vehicle via Bluetooth. The Raspberry Pi is hosting a Python application with four distinct functionalities based on the following libraries: smart contract interaction (*Web3.py*); visualization (*gpio lib*); Bluetooth interaction (*pybleno*); and HSM interaction (wrapper for host library). The HSM host library is written in C, therefore a Python wrapper is required. The host library was auto-generated with an RPC framework introduced in [10]. With that framework, extensive development costs are saved, by just specifying the header file instead manually writing the entire host library. The smart contract is written in Solidity and published with *Infura* onto the *Ropsten Testnet*, which is typically used for development.

C. Procedure

The procedure starts on the smart phone with scanning for Bluetooth devices in the vicinity. The Raspberry Pi in the toy car is broadcasting the preset UUID with a 100ms interval. When selecting the corresponding Bluetooth device in the smart phone application, a write request, including the public wallet address of the user, is triggered. In the demo application the data is send in plain, but a simple symmetric encryption scheme shall be utilized for further use.

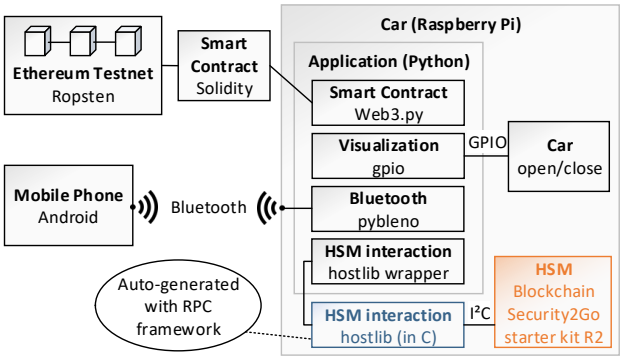

Fig. 7. Demonstrator architecture

With the write request, the *CheckPermission* function of the smart contract is triggered by the Raspberry Pi. If a success message is returned, the transaction as described in Sec. IV-C, is built. Afterwards the HSM signs the transaction with the private key preserved within the HSM. Then, the transaction is transmitted to the *Ropsten Testnet* and if signed successfully included into the next block.

In the successful case, the transaction hash is returned and the wing doors of the toy car are opened. In the blockchain network, the successful unlocking procedure results in sending the vehicle token to the vehicle's wallet address itself with changed parameters lock state and current user.

VI. Evaluation

A. Threat Model

Considering the entire solution and the document [11], where a structured research on the attack surface of blockchains is conducted, the threat model is split into three attack targets: blockchain structure; peer-to-peer system; and blockchain applications. The first two attack targets are out-of-scope of this work, since the focus is on the application. Following the structured research in [11], the most influencing attacks for blockchain applications are: wallet theft; double-spending; and replay attacks, since they result in revenue loss, theft, and info loss.

The focus of this work is to prevent wallet theft, even though countermeasures for double-spending and replay attacks are also in place. The latter attack is no longer a problem on Ethereum based blockchains, since the Ethereum Improvement Proposal (EIP) 155 and the hard fork *Spurious Dragon*, where the transaction hash also includes the chain id as stated in [12]. Double-spending is avoided, since during the unlocking procedure the lock state of the vehicle is validated via the smart contract. Before a vehicle is possible to unlock again, it has to be locked first.

The measures against wallet theft are explained during the comparison to the state-of-the-art solutions.

B. Comparison

In this sub-section the proposed concept is evaluated against state-of-the-art solutions for various aspects, such as security,

TABLE II
STRUCTURED COMPARISON OF STATE-OF-THE-ART SYSTEMS LISTED IN RELATED WORK (SEC. III)

	HireGo [6]	DAV [7]	ZoKrates [4]	Indy [4]	Two token types [6]	Cryptober [8]	velink
Blockchain	Ethereum	Ethereum	Ethereum	Hyperledger	Ethereum	not specified	Ethereum
Token type	ERC-20 (pay) ERC-721 (car)	ERC-20	none	none	ERC-20 (reward) ERC-721 (car)	none	ERC-721
Vehicle (token) ownership	Tenant	Vehicle	-	-	Vehicle	-	Vehicle
Key protection	SW	SW	SW	SW	SW	SW	HSM
Maturity level	Alpha version	Demonstrator	Concept	Concept	Demonstrator	Concept	Demonstrator

complexity, and flexibility. In Table II, a structured comparison is conducted.

The utilized vehicle token is representing the vehicle's digital twin with respective parameters. Only required parameters for demonstration are implemented in the proof-of-concept, but they are extensible. To prevent miss-use of the vehicle, the security perspective is crucial, therefore a state-of-the-art HSM is integrated to protect the key material required for the identity control. Other options such as cloud wallets, hot wallets, or software wallets are available, but compared to hardware wallets they are vulnerable against viruses and other attacks [13], even though the blockchain network itself is tamper-proof by design. In [13], attacks and threats against various wallets are analyzed, and hardware wallets as dedicated cryptographic devices are considered as the most secure.

Another important aspect is complexity. While the proposed concept is only incorporating essential parties, other concepts make the system more complex than required. One system proposed in [4], is intending to implement a membership provider, which is not necessary to be operational. Unnecessary complexity is also introduced in [6], where unlock tokens need to be bought first, instead of directly using ETH. *HireGo* brings the design disadvantage of transferring the vehicle token to the tenant, which results in an ownership loss and hence miss-use is eased [6].

Considering the system explained in [8], tenant and owner need to be in the same place, which lacks in flexibility in regard of the renting procedure. Another promising approach for the future is the *DAV* network, which is trying to lay the foundations for a decentralized transportation infrastructure, focusing on autonomous vehicles [7]. An open-source approach is utilized, but for the shared mobility use case, no details regarding the renting procedure are available.

Further, an evaluation of the setup phase for each listed solution was attempted, but they are not clearly specified in the related resources.

VII. CONCLUSION AND FUTURE WORK

In this work we proposed an HSM supported, blockchain-based vehicle sharing platform leveraging non-fungible ERC-721 tokens. Besides focusing on the interaction between tenant and vehicle, the security perspective is essential. The evaluation has shown, that the proposed solution has an extensive advantage in regard of security. In the proof-of-concept the feasibility was proofed in order to extend the implementation and cooperate with potential partners.

Next steps are to implement the payment and privacy enhancement procedures into the current available proof-of-concept. Further, adopting the system to public transportation and also to the novel flying taxi approach will be analyzed.

VIII. ACKNOWLEDGMENT

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826610. The JU receives support from the European Unions Horizon 2020 research and innovation programme and Spain, Austria, Belgium, Czech Republic, France, Italy, Latvia, Netherlands.

REFERENCES

[1] "Smart Contracts and Solidity," [Online; accessed 2020-08-20]. Available: https://github.com/ethereumbook/ethereumbook/blob/develop/07smart-contracts-solidity.asciidoc#what-is-a-smart-contract

[2] M. d. Angelo and G. Salzer, "Tokens, Types, and Standards: Identification and Utilization in Ethereum," in *2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, 2020, pp. 1–10.

[3] "ERC-721," [Online; accessed 2020-08-26]. Available: http://erc721.org/

[4] I. Gudymenko, A. Khalid, H. Siddiqui, M. Idrees, S. Clau, A. Luckow, M. Bolsinger, and D. Miehle, "Privacy-preserving Blockchain-based Systems for Car Sharing Leveraging Zero-Knowledge Protocols," in *2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, 2020, pp. 114–119.

[5] G. Tripathi, M. Abdul Ahad, and M. Sathiyanarayanan, "The Role of Blockchain in Internet of Vehicles (IoV): Issues, Challenges and Opportunities," in *2019 International Conference on contemporary Computing and Informatics (IC3I)*, 2019, pp. 26–31.

[6] V. Valatn, K. Kotl, R. Bencel, and I. Kotuliak, "Blockchain Based Car-Sharing Platform," in *2019 International Symposium ELMAR*, 2019.

[7] N. T. Copel and Ater, "DAV White Paper," Tech. Rep., 2018, [Online; accessed 2020-08-17]. Available: https://dav.network/whitepaper.pdf

[8] V. Hassija, M. Zaid, G. Singh, A. Srivastava, and V. Saxena, "Cryptober: A Blockchain-based Secure and Cost-Optimal Car Rental Platform," in *2019 Twelfth International Conference on Contemporary Computing (IC3)*, 2019, pp. 1–6.

[9] M. Harikrishnan and K. V. Lakshmy, "Secure Digital Service Payments using Zero Knowledge Proof in Distributed Network," in *2019 5th International Conference on Advanced Computing Communication Systems (ICACCS)*, 2019, pp. 307–312.

[10] T. Fischer, C. Lesjak, D. Pirker, and C. Steger, "RPC Based Framework for Partitioning IoT Security Software for Trusted Execution Environments," in *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2019.

[11] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, and D. Mohaisen, "Exploring the Attack Surface of Blockchain: A Comprehensive Survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1977–2008, 2020.

[12] "EIP-155," [Online; accessed 2020-09-18]. Available: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-155.md

[13] H. Rezaeighaleh and C. C. Zou, "New secure approach to backup cryptocurrency wallets," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.