2019 IEEE Intelligent Vehicles Symposium (IV)
Paris, France. June 9-12, 2019

A Novel Embedded Platform for Secure and Privacy-Concerned Cross-Domain Service Access

Alexander Rech[1], Markus Pistauer[2], and Christian Steger[3]

Abstract— **Connected driving is a hot topic in the automotive industry and a leverage to push new Mobility as a Service (MaaS) methodologies, making vehicles an essential part of the Internet of Things (IoT). However, these new technologies often lead to security risks and privacy concerns, especially due to the increasing number of datasets exchanged between vehicles, drivers, and local infrastructure. Furthermore, the possibilities for vehicles to access heterogeneous services offered by different service providers are often limited due to rigid system boundaries. In this paper we present a novel federated service management concept for increased interoperability across distinct services in the field of Smart Mobility and Smart Cities. Our approach provides secure authentication and authorization between cars, their drivers, and other information systems, while retraining the level of privacy according to the users' preferences. The scalability and dynamic configurability of the solution and the elaborated proof-of-concept will set it apart from application-centered gateways to an embedded generic platform by virtue of its modular software design.**

I. Introduction

In addition to different car-to-car communication concepts car-to-x methodologies have now also become increasingly important. New smart mobility approaches are being elaborated, fueled by the increasing need of new soft- and hardware methodologies and stronger interrelationships between different services. In the automotive sector we can also find more and more subsystems exchanging data with the outside world. As a result, cars are becoming "things" in the IoT and thus also a part of a Smart City, where many different information systems come together. However, more data exchange between a vehicle and the local infrastructure also implies more concerns in terms of security and privacy. Moreover, systems today tend to be tailored to a specific application area, with the result that boundaries are often rigid and inflexible preventing the building of synergies between heterogeneous service providers, their services and users. We now face a critical need for innovations, as secure authentication and authorization to different heterogeneous Smart City services as well as personalized data access become more and more important for cross-domain use cases, especially in the fast growing car-to-x sector.

Personalized access. In this paper we propose a concept for managing user- and service-related data in automotive application scenarios and address the lack of privacy control. The concept enables drivers to adapt the data flow during their driving session according to their privacy preferences. The more permissive they are, the more services may be unlocked, tailored to the driver's data.

Connected services. We call into question the current situation where systems are limited to a predefined scope regarding their services and users. Based on the privacy-centered approach, an additional federated service concept is introduced for accessing and redeeming heterogeneous Smart City services according to the users' privacy settings. In this sense, services of different providers shall be accessible via a trusted cloud-based approach.

Overcoming the offline transition phase. In order to overcome the current non-connected, internet-less transition period in the automotive sector, the communication interface of the embedded proof-of-concept includes the usage of the proximity-based wireless technology Bluetooth Low Energy (BLE). On one hand, it is used to connect the vehicle to the driver's mobile phone which acts as gateway to the cloud via a RESTful interface. The phone is responsible for the initialization process of the hardware including the actual user binding and adaption of service- and privacy-related data. On the other hand, BLE is utilized to enable communication to local infrastructure for redeeming services obtained.

Embedded proof-of-concept. The proposed software concepts were integrated into a hardware module which fulfills the demand for increased flexibility, robustness and configurability. The embedded demonstrator reflects the ongoing trend for property sharing – in our case car sharing – and focuses on personalized access. Additionally, the framework enables the vehicle to access other services, related to parking, ordering food (drive-in-restaurant), or leisure events (cinema, museum), for instance. Our work shows that a connected unit inside a car offers possibilities to bring the connected Smart City paradigm to the automotive sector and thus contributes to a new level of connected mobility within cities, while preserving end user privacy aspects.

This paper is organized as follows. Section II shows related work while section III discusses the design choices of our approach. Section IV consists of implementational aspects and a performance evaluation. Finally, section V summarizes our ideas and gives information on future work.

II. Related Work

Everyday life is becoming ever more connected to the digital world. Not only are information services and applications running on an extensive set of different systems and

[1] Alexander Rech is with CISC Semiconductor GmbH, Graz, 8010, Austria and Graz University of Technology, Department of Technical Informatics, Graz, 8010, Austria `a.rech@cisc.at`

[2] Markus Pistauer is with CISC Semiconductor GmbH, Klagenfurt, 9020, Austria `m.pistauer@cisc.at`

[3] Christian Steger is with Graz University of Technology, Department of Technical Informatics, Graz, 8010, Austria `steger@tugraz.at`

978-1-7281-0560-4/19/$31.00 ©2019 IEEE

are in constant interaction with people and their environment, also the number of wireless connections is growing exponentially around the globe. It is estimated that nearly 25 billion devices will be connected to the internet by 2020 and 50 billion devices by the year 2050. Most of these devices will offer wireless communication interfaces leading to the expansion of wireless areas and the development of novel wireless methodologies [1]. New wireless concepts and frameworks are being developed, such as DEWI. This proposes a locally adaptable and trusted wireless communication bubble for the IoT, where many wireless communication standards are integrated [2]. API-based approaches are a common strategy to interconnect different IoT systems, spanning from access control and area network systems to complex systems that bring together different vendors' solutions. This can be done by connecting IoT endpoints to IoT platforms or gateways which enable the transport of data between multiple devices that would otherwise be unable to communicate with each other. Due to the widespread use of smartphones and their wireless interfaces (Wi-Fi, Cellular, BLE, NFC) they can be used as gateways to connect devices to the internet. BLE is an especially good candidate when power-constrained embedded devices come into play. The work illustrated by paper [3] presents a smartphone-based gateway solution responsible for retrieving data from wearable sensors over BLE as well as storing it to a central cloud storage in real-time. An even more generic gateway approach is shown by the project fabryq presented in the paper [4]. It addresses the problem of increased complexity which arises when developing systems responsible for establishing communication between embedded devices and servers over gateways. By contrast, other approaches involve mobile phones to enable offline communication to cars for secure access in car-sharing application scenarios, by leveraging the phones' local wireless capabilities [5]. Other mobility concepts focus on connecting different e-vehicle solutions. The Horizon 2020 project STEVE for instance, designs and implements a system for connecting different mobility and ”gamified” services [6]. Complementary to this approach is this work [7], which focuses on a more generic method for interconnecting different Smart City services via a trusted cloud-based concept. While security considerations are not new in the context of connected systems, many implementations present new security challenges. Every poorly secured device or subsystem that is connected online can serve as a potential entry point for cyber-attacks, compromising systems as well as exposing data [8]. Slack security methodologies can have especially severe consequences in the automotive sector. Since this sector is becoming increasingly connected (75% of European cars will be connected by 2020 [9]), security vulnerabilities can lead to unsafe or even lethal scenarios, as described in [10], [11]. An increased level of connectivity may not only lead to security problems, but may also evoke major privacy concerns for drivers. As a result, a reluctant connected vehicle user / buyer group of 25% will persist in the next few years, according to the survey ”My Car My Data” [12], leaving one out of four cars unconnected. This makes paving the way for robust, dependable and trustable transport systems all the more important. The paper [13] analyzes how personal information flows through typical telematics systems, distinguishing between an embedded approach where cars connect directly to the internet and an integrated approach relying on mobile devices for accessing different services. Connected cars are able to receive, process, and send large amounts of data. Since this data may not only be related to the vehicle, conclusions about the drivers and their habits (e.g. their destinations, daily routines, etc.) may be derived. Consequently, this causes potential privacy implications for drivers, especially when data is aggregated from several data-sets and shared across various companies, such as car dealers, car rental companies, insurance companies or mobile device and service providers [14]. For a higher level of trust, privacy preserving data management techniques have been studied extensively [15]. New techniques are being elaborated that use clustering algorithms as a pre-process to further improve the diversity of anonymized data [16].

III. Design Choices

A. Distributed system key components

The main objective of our proposed generic solution is to enable cross-domain sharing of heterogeneous services for the driver of a vehicle in a secure and privacy-preserving way. The general idea of the framework is to derive anonymized tokens (authentication-, privacy- and service-tokens) from user and service data and manage their lifecycle (creation, storage, and transfer) among all entities. A custom public key infrastructure (PKI) forms the basis of the platform's security and ensures the identity of all communication participants. Signatures are created and verified with the Elliptic Curve Digital Signature algorithm (ECDSA). The distributed system design, partly based on the architectural proposal of [7], is depicted in Fig. 1 and explained in the following. The elaborated embedded demonstrator inside the vehicle will be referred to as “eClient” in this paper.

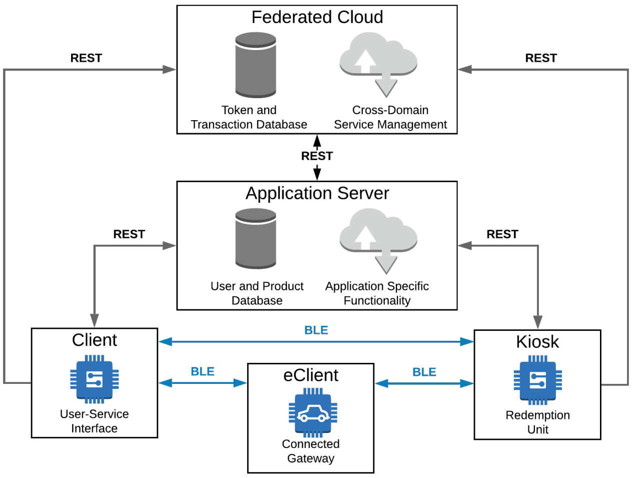

Fig. 1. Distributed system overview.

- **Client.** A mobile personal device representing a user who owns digital tokens and exchanges them for goods and services at corresponding kiosk devices via BLE. It acts as a digital service-wallet and offers an interface for obtaining services and managing the user's privacy rules.
- **Kiosk.** A mobile or stationary device that represents a terminal that can also be embedded into infrastructure (e.g. a parking gate). A kiosk device acts as a validation authority. It receives and validates tokens in order to authenticate users and authorize them to access goods and services.
- **eClient.** An embedded unit that can be placed inside a vehicle to enhance connectivity to its surroundings. It uses a client device without the need of an active internet connection to synchronize user and service data over BLE. Additionally, it is able to redeem the obtained services at kiosk devices.
- **Application server.** This server communicates with client and kiosk devices and keeps them synchronized. It offers application specific functionality and holds the real user data (e.g. email, name, etc.), as well as product and service related datasets (e.g. type, price, etc.) on application level.
- **Federated Cloud.** This works as a certification authority (CA) and token management system for issuing, signing, and monitoring tokens and maintains a transaction record database. The server's certificate is signed by an external root CA and is imported in all system entities' trust CA stores, meaning it is globally trusted. Client and kiosk devices use public key pinning for the root certificates to prevent the issuance of malicious server certificates by tampered CAs. The idea behind the Federated Cloud is to provide a common trusted layer responsible for abstracting users, products, and services from different application servers and sharing the anonymized datasets across the participating systems. In this sense, the real data always remains on application layer level, while only tokenized data is processed inside the Federated Cloud. Different application layers (incl. application server and corresponding apps) of independent service providers may join the Federated Cloud via a RESTful interface and offer their services to other service providers or users.

B. *Wireless communication interfaces*

The overall system provides personalized and secure access to goods and services wirelessly via BLE and a RESTful interface. Which communication interface is used by which device is also depicted in Fig. 1.

- **Representational State Transfer (REST).** REST is a software architectural style for implementing web services relying on HTTP. Client and kiosk devices as well as their corresponding application server communicate to the Federated Cloud via a RESTful API.
- **Bluetooth Low Energy (BLE).** BLE is a short-range wireless technology. It greatly benefits IoT applications due to its power saving design, the coexistence of connectionless (broadcaster and observer roles) and connection-based (peripheral and central roles) data transfer procedures, its robustness against obstacles, and compatibility with smartphones [17]. In our case BLE is used in two different ways, enabling local communication without the need of an active internet connection. On the one hand, it is utilized for the initialization process of the eClient (peripheral) by communicating to the mobile client (central), thus synchronizing user- and service-centered data between both devices. On the other hand, it is used for the actual transaction and redemption handling between a client/eClient (central) and a kiosk device (peripheral).

C. *Secure authentication*

Authentication between devices as well as data-integrity are provided through **authentication tokens (A-tokens)** and a dedicated challenge-response protocol. An A-token is an extended certificate, tied to a particular device. It is server-signed and consists of the device's public key, a validity period, and token properties. A client or kiosk device's A-token is issued in the course of a distributed Kerberos-based authentication procedure involving a user-login on the application server. A one-time-ticket is issued that can be redeemed together with the device's public key during the registration procedure at the Federated Cloud. Finally, the device receives an A-token as well as an API key for accessing the Federated Cloud's REST-based interface.

In order to check the right ownership of A-tokens, a challenge-response procedure is applied when two devices communicate with each other via the BLE channel. First, A-tokens (in this example A-token$_{\text{Alice}}$ and A-token$_{\text{Bob}}$) are exchanged and verified with the server's public key. Alice generates a random number (challenge C_{A}) and challenges Bob to sign it with his private key before it is returned to Alice. Furthermore, the received signature is verified against the random number C_{A} by using Bob's public key embedded into the previously exchanged A-Token$_{\text{Bob}}$. If the verification is passed, the ownership of A-Token$_{\text{Bob}}$ was proven. In case, both devices are already initialized and thus, in possession of an A-token, the same procedure is also applied for A-token$_{\text{Alice}}$ before further data is exchanged. See Fig. 2 for more details regarding the challenge-response mechanism.

Since all packages passed between the devices do not contain any sensitive data but only tokenized user and service datasets, an additional encryption layer is not required. This implies benefits such as less computational efforts for all communication participants and faster data transmission.

D. *Adaptive authorization of different services*

We distinguish between two generic methodologies when talking about authorization. On the one hand, our concept foresees the usage of user-centric **privacy-tokens (P-tokens)**, which manage the driver's privacy level. On the other hand, service-related **service-tokens (S-tokens)** represent a digital ownership of an item or service. They are issued

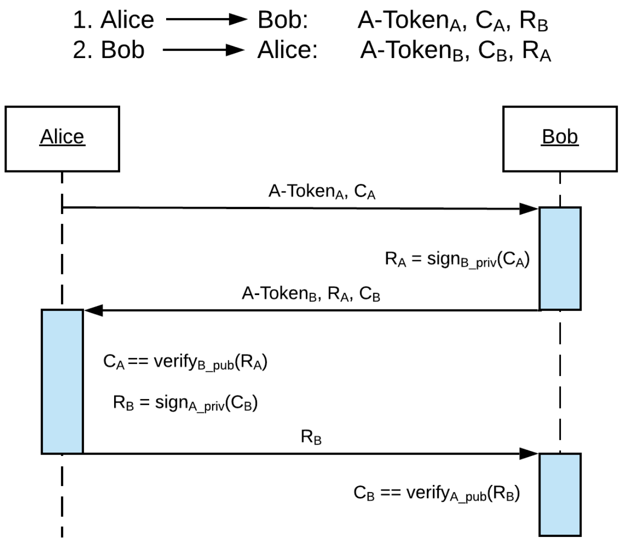
1. Alice → Bob: A-Token$_A$, C_A, R_B
2. Bob → Alice: A-Token$_B$, C_B, R_A
Alice
Bob
A-Token$_A$, C_A
R_A = sign$_{B\_priv}$(C_A)
A-Token$_B$, R_A, C_B
C_A == verify$_{B\_pub}$(R_A)
R_B = sign$_{A\_priv}$(C_B)
R_B
C_B == verify$_{A\_pub}$(R_B)

Fig. 2. A-token based challenge response mechanism.

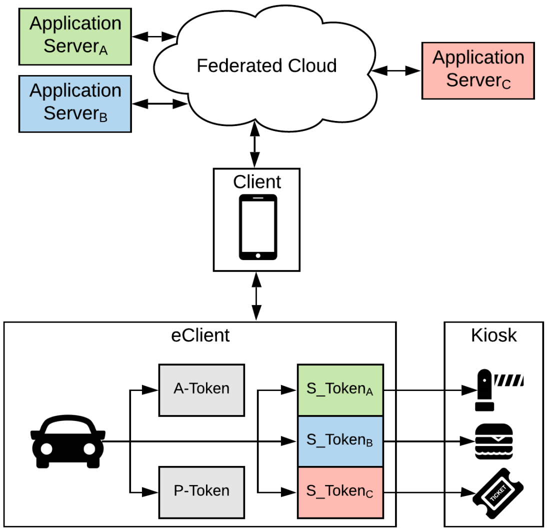

Fig. 3. Overview of application scenarios.

the moment a user obtains a product or service voucher via the application interface of the client. Both token types are always issued and signed by the Federated Cloud, bound to a specific A-Token, and only valid when presented in combination with the latter.

Furthermore, the S-Token consists of a validity period and two additional identifiers: one for the corresponding application (e.g. parking application, drive-in-restaurant application) and the other one for the actual service (e.g. right to access a specific area). They serve as indicators to specify which S-tokens are redeemable by a specific kiosk device. If these conditions are met, the appropriate S-token can be sent from a client or eClient device to a kiosk and be redeemed there. Consequently, the user is authorized to access the service for which the tokens stands. Due to the generic nature of an S-token, services of different kinds can be represented, thus enabling different use cases. An overview of a few possible application scenarios is given in the following. A pertinent schematic description can be taken from Fig. 3.

- **Car access.** As soon as an A-token and a corresponding P-token have been transferred to the eClient, thus authenticating a user and binding himself and his privacy settings to the eClient, the vehicle's unlocking signal can be triggered. Since the transfer of the tokens does not involve an active internet connection on the eClient side, and each user is able to personalize his dataflow during the trip with his personal device, this would suite a car sharing use case. Additionally, all data on the eClient could be reset after the car-sharing session.
- **Smart parking.** When a valid parking entitlement is transferred to the eClient, the vehicle is allowed to access a parking lot for the validity period of the entitlement.
- **Drive-in-restaurant.** If a food voucher is obtained, the vehicle will be able to redeem the food voucher directly at the kiosk of a drive-in-restaurant.

By contrast, a P-token contains the datasets the user wants to share as well as the user's privacy level which can be edited via the application interface of the Client. The P-token is transferred from client to eClient during the initialization process. The data that can be shared may include user-related data like gender or age, or trip-based datasets collected during the car ride, such as information about the destination, GPS coordinates, etc. depending on the underlying application. If the user decides upon a low privacy level and thus consents to the data in question being shared with other service providers, it is forwarded to the Federated Cloud and can be used in exchange to issue additional S-tokens for the driver. The uploaded data is not directly associated with privacy critical values such as his name or email-address, but just linked to his anonymized P-token. In order to cope with different privacy requirements three privacy levels were introduced:

- Level 2: Only services obtained directly by the user are accessed. No additional data is shared across the Federated Cloud.
- Level 1: Drivers may decide whether trip-based (e.g. GPS, destination) or user-related datasets (e.g. gender, obtained services) are shared.
- Level 0: User and trip-based data collected during the journey are shared with participating service providers.

This approach foresees that the application server will utilize the RESTful interface to send key/value pairs to the Federated Cloud containing the data in question. Subsequently, the values submitted are linked to the user's P-token. They become visible to other participating service providers for as long as the user does not change his privacy settings or his A-token or P-token do not expire. In further consequence, the datasets submitted may be used to create new offers and entitlements adapted to the user's needs and preferences, involving again the Federated Cloud.

In this context, the use cases mentioned above can be arbitrarily extended, like in the following example:

Generic interaction with heterogeneous service providers. In cases when data is shared, service providers may offer their users additional entitlements for accessing their services for special conditions. In concrete terms, a car wash agency (e.g. Application Server$_C$ in Fig. 3) would be able to access shared data over the Federated Cloud without knowing the real identity of the driver. For example, when the driver's destination address is shared, the car wash agency could offer him a discounted or free admission to a car wash session lying on the route to his destination. Consequently, the derived S-token$_C$ would be stored on the user's eClient and is redeemable at a dedicated kiosk device at the car wash.

IV. Implementation And Results

A. *eClient components*

The eClient consists of two complementary components, the Adafruit Feather M0 board and the Bluegiga BLE112 module. While the Feather M0 acts as control unit and state machine, the main task of the BLE112 module is to execute different BLE commands and forward the responses received back to the Feather M0 over the UART interface. The Feather M0 is based on the widely spread Arduino platform. It was picked as host device since the implementation should remain as portable as possible for similar controllers. The following software libraries were included into the project:

- **BGLib.** This library acts as a C wrapper for the event-driven BGLib protocol used to control the Bluegiga BLE112 module.
- **micro-ecc.** This library holds a lightweight ECDH and ECDSA implementation for 8-bit, 32-bit, and 64-bit architectures. In our case, the elliptic curve secp256r1 (prime256v1, NIST P-256) was used. According to the 2018 ECRYPT-CSA recommendation this type of curve is recommended at least until 2028. The major advantage of ECDSA is its short cryptographic key length compared to other algorithms such as RSA. This enables faster data transfer of keys and certificates and lower memory requirements.
- **Cryptosuite.** It is a cryptographic library specialized on secure hashing and hashed message authentication. SHA-256 was used in our case.

The eClient provides possibilities to bring the connected services paradigm to vehicles, without providing a direct interface to the car. The small form factor of the Feather M0 (2in x 0.9in) combined with the possibility of attaching a rechargeable lithium polymer or lithium ion battery over the JST jack makes it even more portable.

B. *Mobile client as gateway*

The client acts as an internet-enabled gateway for the eClient. It was implemented in the form of a Java-based software library which can be integrated into Android applications. Regarding Android's BLE stack all BLE roles are supported since Android 5.0. The additional possibility to only scan for specific BLE advertisements was also introduced with version 5.0. All interactions that would require the eClient to directly communicate to the Federated Cloud are carried out by the mobile client instead. In this sense, the eClient sends out specific BLE advertisements which are noticed by the client. Subsequently a BLE connection is established and the actual request is forwarded to the mobile client and finally, carried out using the REST interface of the Federated Cloud. Authentication to the Federated Cloud is ensured by providing a secret API key for basic HTML authentication in the request, which is issued during the registration of the device. Next, the server's response is sent to the client, processed, and forwarded to the embedded device over BLE. The server response is formatted in JSON while all data transferred between the eClient and the gateway is encoded in the more compact TLV-encoding scheme, conforming to the following byte format: Type | Length | Value. This format allows the receiver to decode the information with dedicated parsing functions without requiring any pre-knowledge of the size or the semantic meaning of the data.

C. *Key and token handling*

As soon as the eClient is switched on the initialization process is started and is responsible for setting up the BLE module and the board's pins, and checking if the cryptographic keys and corresponding A-token or P-token are already stored on the device. If the device still needs to be initialized, an ECC key pair ($k_{priv\_eClient}$, $k_{pub\_eClient}$) is generated and stored. The elliptic curve secp256r1 is used for signing and verifying signatures. Therefore, the private key is 32 bytes and the public key 64 bytes long. As soon as a user approaches the vehicle with his mobile client a confirmation log will appear. If confirmed, a BLE connection between client and eClient is established and the binding procedure is triggered. First, the eClient verifies the authenticity of the mobile client:

1) The client sends its device and user-bound A-token$_{client}$ to the eClient.
2) The expiry date of A-token$_{client}$ is checked by comparing the current date and time with the timestamp value inside the token.
3) The signature of A-Token$_{client}$ is verified with k_{pub_server} before a challenge response protocol (see section III-C) is applied to verify if the A-token received really belongs to the device currently communicating to the eClient.

The next steps involve the derivation of new tokens for the eClient from the user-bound A-Token$_{client}$ and the privacy settings in form of a P-Token$_{client}$.

4) The eClient sends a data packet containing $k_{pub\_eClient}$ to the client, which will forward it together with its tokens to the Federated Cloud.
5) The server creates a derived A-token$_{eClient}$ and P-token$_{eClient}$ where $k_{pub\_eClient}$ is embedded.
6) The new tokens are sent back to the smartphone client and forwarded to the eClient through the BLE-channel.
7) Both tokens received are verified with k_{pub_server} by the eClient.

Next, a unique fingerprint (A-token$_{\text{client\_fingerprint}}$) of the mobile client is created by hashing A-Token$_{\text{client}}$ with SHA-256. For further communication this fingerprint is used together with the previously utilized challenge-response-protocol to determine if the same device responsible for the initialization mechanism is communicating to the eClient. Finally, A-Token$_{\text{client\_fingerprint}}$, A-Token$_{\text{eClient}}$, and P-Token$_{\text{eClient}}$ are stored on the eClient. From this moment on S-tokens can also be synchronized between the two devices involving again the Federated Cloud for the derivation process. Last but not least, the digital services obtained can be redeemed at corresponding kiosk entities.

D. Performance evaluation

The following three tables provide information about the timing behavior of the implementation. Average measurement values are reported.

Table I shows the time the token generation process for the eClient takes. It includes the A-Token$_{\text{eClient}}$ and P-Token$_{\text{eClient}}$ derivation involving the client and the Federated Cloud and several verification steps.

TABLE I

MEASURED TIME OF THE A-TOKEN$_{\text{ECLIENT}}$ AND P-TOKEN$_{\text{ECLIENT}}$ DERIVATION PROCEDURE

Action	Time [ms]
Retrieve and verify A-Token$_{\text{client}}$	388
Derive and receive A-Token$_{\text{eClient}}$ and P-Token$_{\text{eClient}}$	1210
A-Token$_{\text{eClient}}$ and P-Token$_{\text{eClient}}$ verification	421
eClient local data-storing procedure	73
Total	**2092**

In contrast, the information on the time required for the verification mechanism between eClient and Client as well as the creation and reception of an S-token$_{\text{eClient}}$ is given by Table II.

TABLE II

MEASURED TIME OF THE S-TOKEN$_{\text{ECLIENT}}$ DERIVATION PROCEDURE

Action	Time [ms]
Mutual verification mechanism	1288
Derive and receive S-Token$_{\text{eClient}}$	442
eClient local data-storing procedure	59
Total	**1789**

Last but not least, Table III illustrates the timing behavior of the verification and data transfer procedure between an eClient and kiosk device, and the online redemption process of an S-Token$_{\text{eClient}}$ involving the Federated Cloud.

TABLE III

MEASURED TIME OF THE REDEMPTION PROCESS BETWEEN ECLIENT AND KIOSK DEVICES

Action	Time [ms]
Mutual verification mechanism	1194
Prepare data to redeem	533
S-Token$_{\text{eClient}}$ online redemption	452
Total	**2179**

In summary, the retrieval of the tokens and the redemption process take around two seconds with the verification mechanism taking a large part of the total execution time. However, since the developed prototype does not have any realtime requirements, the overhead created by the verification mechanism is still acceptable. The values represent short waiting times for the user underlining the usability aspect of the developed prototype.

V. CONCLUSION AND FUTURE WORK

With the ongoing development and distribution of the Internet of Things, new car-to-x methodologies have also become increasingly important. In order to fuel the movement towards MaaS we designed and implemented an embedded prototype that gives cars and their drivers access to heterogeneous services (car access, smart parking, drive-in restaurant food-voucher redemption, etc.), while respecting the drivers' privacy requirements. In view of the number of not internet-connected-cars that is still not negligible [9], [12], our proof of concept takes advantage of the local wireless communication standard BLE to overcome the current non-connected transition phase in the automotive sector. In this sense, it uses the driver's mobile phone as a gateway for a coupling procedure between car and driver and the synchronization of the tokenized user- and service-related data. Furthermore, the device is able to redeem obtained services by communicating to local infrastructure (e.g. parking gates). Secure authentication and authorization as well as data integrity are enforced via cryptographic standards such as PKI and ECDSA in order to meet the high demands on security and privacy. Additionally, all data transmitted between the devices is issued, tokenized, and monitored by a central trusted server. Due to the generic nature of the tokens, services of multiple heterogeneous service providers can be represented, offering the possibility to arbitrarily extend MaaS application scenarios (e.g. drivers get a digital ticket for a car wash lying on their route). Finally, the evaluation of the transmission timings revealed low protocol execution rates – on average of approximately two seconds – thus highlighting the prototypes overall usability. Future Work will concentrate on defining a secure interface between the embedded proof of concept and the vehicle's on-board unit. In this way, additional services may be unlocked for the driver of the vehicle, due to the larger number of available data. Furthermore, we intend to increase the level of trust between participating service providers with a distributed smart-contracts-based approach.

ACKNOWLEDGEMENT

This work is supported by CISC Semiconductor and Graz University of Technology (Institute of Technical Informatics). Furthermore, parts of this work have been performed within the European Union's Horizon 2020 projects SCOTT under agreement No. 737422 and PRYSTINE under grant agreement No. 783190.

REFERENCES

[1] D. Evans, "The Internet of Things - How the Next Evolution of the Internet is Changing Everything," *CISCO white paper*, pp. 1–11, 2011.

[2] W. Rom, P. Priller, J. Koivusaari, M. Komi, R. Robles, L. Dominguez, J. Rivilla, and W. V. Driel, "DEWI-Wirelessly into the future," in *Proceedings - 18th Euromicro Conference on Digital System Design, DSD 2015*, 2015, pp. 730–739.

[3] T. Soultanopoulos, S. Sotiriadis, E. Petrakis, and C. Amza, "Internet of Things data management in the cloud for Bluetooth Low Energy (BLE) devices," *Proceedings of the Third International Workshop on Adaptive Resource Management and Scheduling for Cloud Computing - ARMS-CC'16*, pp. 35–39, 2016.

[4] W. McGrath, M. Etemadi, S. Roy, and B. Hartmann, "Fabryq: Using Phones As Gateways to Prototype Internet of Things Applications Using Web Scripting," in *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS)*, 2015, pp. 164–173.

[5] A. Dmitrienko and C. Plappert, "Secure Free-Floating Car Sharing for Offline Cars," *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy - CODASPY '17*, pp. 349–360, 2017.

[6] The European Commission, "STEVE: Smart-taylored L-category electric vehicle demonstration in heterogeneous urban use-cases." [Online]. Available: http://www.steve-project.eu

[7] A. Rech, M. Pistauer, and C. Steger, "Increasing Interoperability Between Heterogeneous Smart City Applications," in *Lecture Notes in Computer Science*. Tokyo: Springer International Publishing, 2018, pp. 64–74.

[8] K. Rose, S. Eldridge, and L. Chapin, "The Internet of Things: An Overview - Understanding the Issues and Challenges of a More Connected World," Tech. Rep., 2015.

[9] Frost & Sullivan, "The Future of Connected & Autonomous Vehicles," *Intelligent Mobility Report*, pp. 1–12, 2015.

[10] C. Valasek and C. Miller, "Remote Exploitation of an Unaltered Passenger Vehicle," *Technical White Paper*, pp. 1–91, 2015.

[11] C. Miller and C. Valasek, "Adventures in Automotive Networks and Control Units," *IOActive Technical White Paper*, pp. 1–99, 2013.

[12] Fédération Internationale de l'Automobile, "What Europeans Think About Connected Cars," Tech. Rep., 2015.

[13] K. Jaisingh, K. El-Khatib, and R. Akalu, "Paving the way for Intelligent Transport Systems (ITS)," in *Proceedings of the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications - DIVANet '16*. Malta: ACM, 2016, pp. 25–31.

[14] P. Lawson, B. McPhail, and E. Lawton, "The Connected Car: Who is in the Driver's Seat? A study on privacy and onboard vehicle telematics technology," Tech. Rep., 2015.

[15] V. Torra, G. Navarro-Arribas, and K. Stokes, "An Overview of the Use of Clustering for Data Privacy," in *Unsupervised Learning Algorithms*, M. E. Celebi and K. Aydin, Eds. Cham: Springer International Publishing, 2016, pp. 237–251.

[16] P. Canbay and H. Sever, "The Effect of Clustering on Data Privacy," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. Miami: IEEE, 2015, pp. 277–282.

[17] R. Davidson, T. Kevin, W. Chris, and C. Cufí, *Getting Started with Bluetooth Low Energy Tools and Techniques for Low-Power Networking*, 1st ed., B. Sawyer and M. Loukides, Eds. O'Reilly Media, 2014.