SIUV: A Smart Car Identity Management and Usage Control System Based on Verifiable Credentials

Ali Hariri, Subhajit Bandopadhyay, Athanasios Rizos, Theo Dimitrakos, Bruno Crispo, Muttukrishnan Rajarajan

SIUV: A Smart car Identity management and Usage control system based on Verifiable credentials

Ali Hariri[1,2], Subhajit Bandopadhyay[1,3], Athanasios Rizos[1], Theo Dimitrakos[1,4], Bruno Crispo[2], and Muttukrishnan Rajarajan[3]

[1] Munich Research Center, Huawei Technologies Duesseldorf GmbH, Munich, Germany, {name.surname}@huawei.com
[2] Department of Computer Science and Information Engineering, University of Trento, Trento, Italy{name.surname}@unitn.it
[3] Institute for Cyber Security, School of Mathematics, Computer Science and Engineering, City, University of London, London, United Kingdom {subhajit.bandopadhyay, r.muttukrishnan}@city.ac.uk
[4] School of Computing, University of Kent, Canterbury, United Kingdom

Abstract. The automotive industry is witnessing an accelerated growth in digital innovations that turn modern vehicles into digital systems. This makes the security of modern vehicles a crucial concern as they have evolved into cyber-physical and safety-critical systems. Therefore, stateful identity management and continuous access control have become a paramount requirement in smart vehicles. Indeed, several Identity and Access Management (IAM) frameworks have been proposed in the automotive field, but context awareness and continuity of control remain overlooked. To address these challenges, we present SIUV: a stateful smart-car IAM that is based on Usage Control (UCON) and Verifiable Credentials (VCs). SIUV uses Attribute Based Access Control (ABAC) policies to issue privileges to subjects (i.e. drivers or applications) according to their credentials and claims. The issued privileges are then used to decide whether to grant or deny access to in-car resources. Furthermore, the system continuously monitors subject claims, resource attributes and environmental conditions (e.g. location or time). Hence, if a change occurs, the system re-evaluates policies and updates or revokes issued privileges and usage decisions accordingly. We describe the architecture of SIUV, discuss the evaluation results, and define future directions.

Keywords: UCON · IAM · Automotive · Smart Car · Verifiable Credentials · Principle of Least Privilege · ABAC · ALFA · XACML · Ed25519

1 Introduction

The exponential growth of the Internet of Vehicles (IoV) innovations, such as autonomous driving, driver assistance, vehicle connectivity, infotainment and shared mobility has recently gained a notable attention in the automotive industry. Such services and innovations are supported by the integration of smart

sensors as well as the shift from electromechanical to interconnected software-based systems in modern vehicles [6][9]. In addition, vehicular systems are transforming towards centralised architectures in order to support dynamic services and functionalities. Particularly, Service-Oriented Architectures (SOAs) are increasingly being adopted in automotive frameworks, due to their flexibility. SOAs provide a better abstraction and separation between hardware and software, and allow smoother changes and integration of services. However, the flexibility and dynamicity of SOAs comes at the expense of the simple security of the statically predefined functions of existing architectures. Although SOAs are being adopted in automotive architectures, typical SOA security measures are not completely sufficient due to the safety-critical nature of vehicles [23].

The security of modern vehicles has gained more focus in the research community and the automotive industry. For instance, Miller and Charlie [18] performed an attack that allowed them to gain access to critical physical systems such as steering and braking systems. As a result, few millions of cars had to be recalled from the market. Similarly, Dürrwang et al. [12] exploited a vulnerability that allows an unintended deployment or a prevention of deployment of airbags. More recently, Wouters et al. [26] uncovered a vulnerability that allowed them to clone a key fob of a Tesla Model S in less than two minutes. Researchers have also studied the state of the art of automotive security and defined open problems and challenges [23][3]. Dynamic access control and identity management are among the critical challenges that need to be addressed, because vehicles are transforming into digital systems where applications are provided by third party providers. The behaviour of such applications is dynamic and cannot be known in advance, so they cannot be trusted unconditionally. Therefore, such applications must be continuously verified and their internal and external communications and access to resources must be controlled and monitored [23][3][7].

Several researchers have proposed vehicular access control frameworks to address the aforementioned challenges. For instance, Hamad et al. [13] introduced an authentication and authorisation framework, based on a trust management model, that blocks or allows communications initiated by in-car applications Kim et al. [16] developed a decentralised access control framework by integrating an Attribute Based Access Control (ABAC) module in AUTOSAR adaptive platform [2]. Rumez et al. [22] also introduced a distributed ABAC tailored for automotive architectures. Both [16] and [22] focused on protecting Electronic Control Unit (ECU) diagnostic interfaces from unauthorised access. Likewise, Ammar et al. [1] developed an end-to-end Role Based Access Control (RBAC) mechanism that regulates access to On-Board Diagnostics-II (OBD-II) ports.

In spite of the significant contributions of the aforementioned works, context awareness, continuity of control and dynamic identity management remain overlooked. These aspects are crucial because vehicles are real-time mobile systems whose environmental conditions change continuously as they move. Thus, access to vehicular resources must be continuously monitored and controlled to ensure correct, safe and secure usage as circumstances change. The Principle of Least Privilege (PoLP) is also another challenge that need to be addressed in

order to mitigate insider threats and eliminate the risk of unintended or malicious use of unnecessary capabilities. Furthermore, identities and privileges of subjects (i.e. drivers, passengers and applications) need to be continuously managed, monitored and updated according to contextual changes. To address these challenges, we present Usage Control System Plus (UCS+): an optimised, efficient and modular implementation of the Usage Control (UCON) model tailored for embedded and safety-critical systems such as smart cars. More importantly, we introduce SIUV: a vehicular Identity and Access Management (IAM) system that supports dynamic and stateful identity management, context-aware and continuous usage control, as well as the PoLP. SIUV incorporates a centralised stateful and dynamic Security Token Service (STS) that manages, authenticates and verifies identities, and issues privileges in exchange using Verifiable Credentials (VCs). The STS exchanges external VCs that hold identity claims about subjects with internal VCs that enclose subject privileges and capabilities. The STS uses policy-based decision making to that defines how to exchange identity claims with privileges taking environmental conditions into account. It also uses continuous monitoring and policy re-evaluation in order to manage the life-cycle of privileges and adapt to changing situations. Following the trend of adopting SOAs in vehicles, we also propose an SOA-based vehicular IAM architecture that consists of the aforementioned STS in addition to distributed UCS+ instances that protect localised in-car resources. We only focus on protecting internal vehicular resources, but SIUV can be extended to control external communications. To sum up, the main contributions of this work are as follows:

- **Dynamic and continuous authorisation:** optimised, efficient and modular implementation of the UCON model (Section 3.1)
- **Context-aware STS:** policy-based STS that manages a continuous life-cycle of privileges and adapts to changing situations (Section 3.2)
- **SIUV:** SOA-based dynamic and context-aware IAM system (Section 3.3)

The rest of this paper is organised as follows: Section 2 presents the technical background, namely the UCON model, the used policy language, and VCs. We describe UCS+ as well as SIUV architecture and STS in Section 3. An example use case and the evaluation of the system are discussed in Section 4. Finally, we define future directions and draw conclusions in Section 5.

2 Background

This section outlines the theoretical and technical background of SIUV.

2.1 Access and Usage Control

ABAC [14] is one of several access control models used to protect digital resources. It provides more flexibility and finer-grained control than preceding models as it manages access rights based on attributes of subjects, resources

and the environment. ABAC's evaluation semantics, architecture, administration and policy language are defined in a comprehensive standardised reference model known as the eXtensible Access Control Markup Language (XACML) [20]. Although ABAC is fine-grained and flexible, it does not support continuity of access and mutability of attributes during evaluation. For this reason,the UCON model was proposed by Park and Sandhu [21] as a generalisation that goes beyond ABAC and other models to support context awareness, mutability of attributes and continuity of control. UCON continuously monitors attribute values and re-evaluates policies when a change occurs in order to guarantee that access rights still hold whilst usage is still in progress. The model categorises decision predicates as "*pre*", "*ongoing*" and "*post*". "*pre*" predicates are evaluated when an access request is made in order to decide whether to grant or deny access; "*ongoing*" predicates are evaluated during the time span of access, and they decide whether to revoke or retain access; and "*post*" predicates are evaluated after the end/revocation of access. In addition, UCON introduces obligations and advice; such that obligations are actions that must be fulfilled, whereas advice refers to actions that are recommended. The novelties of UCON make it an excellent baseline for dynamic applications, such as IoV, where the context changes continuously and resource usage is long-lived.

2.2 Abbreviated Language For Authorisation (ALFA)

Abbreviated Language For Authorisation (ALFA) [19] is a pseudocode domain-specific policy language that maps directly to XACML without adding any new semantics. It is much less verbose than XACML and thus more human readable and shorter in size. We use ALFA as the baseline policy language of the IAM because its compact size allows faster parsing and evaluation, which is imperative in safety-critical applications like smart vehicles. In addition, a policy-based IAM allows a dynamic and codeless behaviour that adapts according to the context.

2.3 Verifiable Credentials (VCs)

Verifiable Credentials are digital identifiers that provide cryptographic proofs to support the validity and reliability of a claim. The W3C VC data model [25] specifies the roles of claims, credentials, presentations etc. to help conform to a common structure. We construct such claims as privileges, which authorise a subject to use specified car resources. If we consider a VC as a presentation graph, it could consist of different credentials which are linked to each other contextually and packaged together to give a unique presentation by the holder. Access to car resources requires appropriate authorisation and should be controlled through credentials and continuously monitored through policies and usage-checks. A VC is suited to store such credentials because it is cryptographically verifiable and tamper-resistant. The set of claims that a VC makes on behalf of the issuer or multiple issuers can refer to a common or multiple subjects too, thus making the case for multiple identity properties be used for a distinct purpose. The VC information flow allows an issuer to issue credentials to the holder. The

credentials are stored in a wallet and the holder decides whether to present the information in the form of a credential or presentation to the verifier. Every credential or presentation includes a proof mechanism that helps the verifier to verify the authenticity of the VC with the Verifiable Data Registry (VDR). We adhere to the W3C VC model [25] that specifies that every VC must include a proof mechanism to support the verifiability of the credentials. We used Ed25519 [4], a twisted Edwards curve digital signature algorithm, based on elliptic curve cryptography. As of today, Ed25519 is the most popular instance of EdDSA and is based on the Edwards Curve25519. Although there are many variants of Ed25519-original [4] such as NIST [8], IETF [15] etc. that specifies some refined security properties, we use the Ed25519 instance by LibSodium [10], a widely popular cryptographic library. Brendel et al. [5] discusses the game-based definitions of the security properties of the Ed25519 signature scheme and provably defines Ed25519-LibSodium [10] to be more resilient against key substitution attacks as well as message bound security than other Ed25519 instances [4][8][15].

3 SIUV

In this section, we describe the STS in addition to the architecture of SIUV.

3.1 Usage Control System Plus (UCS+)

Lazouski et al. [17] introduced a Usage Control System (UCS) prototype that realises the UCON model. They used a policy language that extends XACML semantics with UCON novelties and defines an architecture to enforce UCON policies. The language adds an implicit temporal state that is captured by classifying policy rules as "*pre*", "*ongoing*" and "*post*". UCS enforces continuous usage control by adding an authorisation session, continuous monitoring of attributes and re-evaluation of relevant policies when a change occurs. However, sessions in UCS are limited to the duration of active authorisations only. This is not sufficient for applications, such as smart vehicles, that include continuous interactions before or after access (e.g.ensure safe release of resources). This was among the motivations for our team to introduce UCS+: an enhanced and optimised UCON framework that conserves a full ABAC baseline and supports auxiliary evaluators like trust/confidence level. UCS+ extends authorisation sessions to cover continuous interactions and monitoring before granting access, during authorisation and during revocation of access in order to support pre- and post-usage interactions such as multi-factor authentication, safe revocation, etc. (e.g. safely stop the vehicle before completely revoking access). UCS+ uses ALFA - instead of XACML - as the baseline policy language, which results in a considerable improvement in performance and efficiency due to ALFA's compactness. UCS+ leverages the publish/subscribe pattern to maximize concurrency between policy parsing and evaluation, and attribute retrieval. This maximizes performance and minimizes the need for high network speeds or high computational resources. UCS+ also improves the ability to upgrade or substitute

component services and migrate to a distributed deployment where necessary. We presented, in [11], a variant of UCS+ that integrates a trust level evaluation engine and is tailored for zero-trust Internet of Things (IoT) networks. In this paper, we describe another variant that is optimised for efficient continuous authorisation in embedded systems including automotive units. Figure 1 illustrates the architecture of UCS+, which consists of 8 major components as follows: `Context Handler (CH)` is the core component that receives access requests and manages authorisation workflows. `Message Bus` supports communications between components using the Publish/Subscribe (Pub/Sub) pattern. `Policy Enforcement Point (PEP)` is the interface of UCS+ and the component that protects resources. It creates access requests, invokes the CH and enforces decisions. `Policy Decision Point (PDP)` is the component that evaluates policies and makes access decisions. `Policy Administration Point (PAP)` stores and manages policies and is used by the PDP to retrieve applicable policies. `Policy Information Point (PIP)` defines where to find attributes and how to monitor them. `PIP Registry` manages PIPs and defines which PIPs are responsible for which attributes. `Session Manager (SM)` manages and keeps track of all ongoing sessions to support the continuity of control. `Attribute Table (AT)` is a cache of attribute values and other metadata. `Attribute Retriever (AR)` is an auxiliary component in charge of querying and updating attribute values. `Obligation Manager (OM)` handles and manages policy and rule obligations.

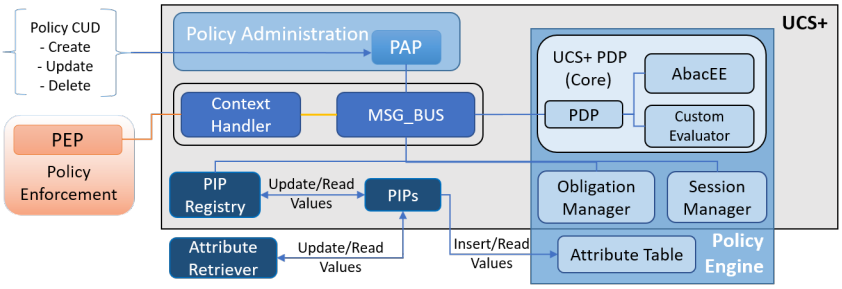

Fig. 1. UCS+ architecture

3.2 SIUV Security Token Service (STS)

In a typical STS, a particular set of privileges may be bound to specific attributes. Thus, all subjects that have these attributes will always have the same privileges regardless of the context. In addition, issued privileges do not change even if attributes change. This usually results in subjects being either overprivileged or underprivileged. To solve such issues, we introduce a stateful policy-based STS that exchanges external VCs about identity claims (identity VCs) with internal contextualised VCs that determine the privileges of the corresponding subject

(privilege VCs). The STS model shares some common principles with UCS+, namely policy-based decision making, session-based continuous monitoring, as well as policy re-evaluation and revision. While UCS+ uses these concepts to manage authorisations, the STS uses them to improve the dynamicity as well as situation and change awareness throughout the life-cycle of identity claims and privileges. The STS employs policy-based decision making in order to dynamically exchange identity claims with privileges according to situation-aware policies. In addition, session-based continuous monitoring allows the STS to manage a continuous lifecylce of privileges starting from their issuance and lasting until their revocation or expiry. The privilege life-cycle may also last beyond revocation in order to support post-revocation interactions such as graceful revocation or safety actions. Finally, policy re-evaluation allows the STS to react to changes in identity claims or environmental conditions, which may result in privilege escalation, degradation or revocation. The architecture of the STS is shown in Figure 2. The PEP receives identity VCs from the subject, verifies them using the Claims Verifier subcomponent, then sends a privileges request to the CH. The CH retrieves relevant policies from the PAP, collects required attributes from PIPs then invokes the SM to create a session and manage the life-cycle of the privileges to be issued. Thereupon, the CH invokes the PDP to evaluate the policies, then returns the evaluation decision to the Privileges Issuer subcomponent of PEP, which issues privilege VCs.

We use ALFA policies to determine how the STS issues privileges according to identity claims and environmental attributes. We particularly use rule obligations to determine the specific privileges that must be issued. Thus, each rule is used to define a set of privileges to be issued if the conditions of the rule are met. This allows a fine-grained control on how to issue privileges according to attribute values. We use the "permitUnlessDeny" combining algorithm, which combines all obligations from all applicable rules that evaluate into permit as long as no deny rule apply. An example policy is shown in Listing 1.1.

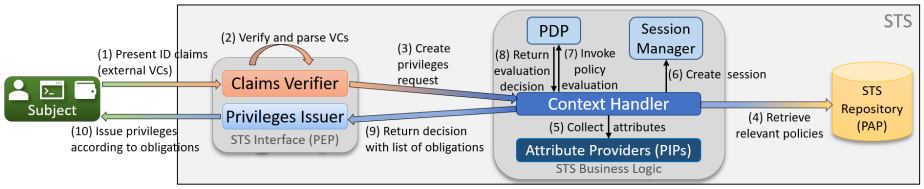

Fig. 2. SIUV STS architecture

3.3 SIUV Architecture

In this section, we present the architecture of SIUV as shown in Figure 3. The STS runs on the general-purpose computer gateway, whereas UCS+ instances

run on the ECUs that they protect. The VDR is a logical component and is implemented as a distributed hashtable. Since Smart-car Identity management and Usage control system based on Verifiable credentials (SIUV) is based on SOA, all communications use the Ethernet protocol. Moreover, AUTOSAR specifies the SOME/IP[1] protocol that supports TLS, thus communications between SIUV components are protected. Furthermore, additional features, such as end-to-end encryption[2], can be used to enhance the protection between components. We describe the components that support dynamic and stateful identity management, and context-aware and continuous usage control of localised in-car resources.

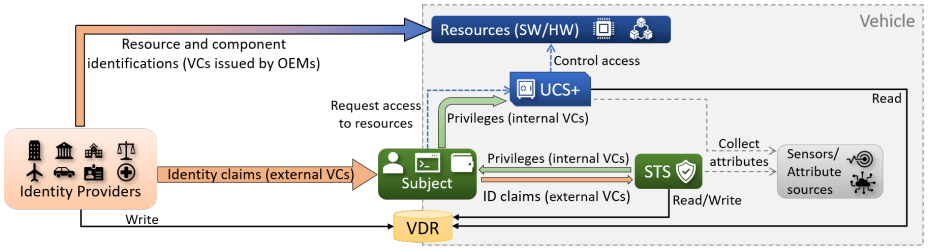

Fig. 3. SIUV architecture

Identity Providers (IdPs) are authorities that issue identity VCs about drivers, passengers, applications or in-car resources. An example of an issuer is the department of motor vehicles that issues driving licenses, or an Original Equipment Manufacturer (OEM) that asserts claims about an ECU.

A subject may be a driver, a passenger, an application or a resource that needs to access other resources, and is represented by a digital wallet that holds the subject's VCs. The digital wallet interacts with the issuers and the STS to present/obtain VCs, and with UCS+ instances to request access to resources.

The VDR is a distributed hashtable that holds revocation lists and issuers' public keys. The VDR may also be used to store other relevant information like metadata about proofs or schemas and structures of VCs.

The STS is the core component that provides dynamic identity and privilege management of subjects. It exchanges external identity VCs with internal privilege VCs. The combination of VCs and the STS protects against identity theft as privileges issued by the STS are cryptographically verifiable and cannot be manipulated. This also allows *unlinkability* and maximizes privacy of subjects because they do not need to share identity information with the OEMs of the vehicle's components. It runs on the centralized general-purpose gateway, which has enough computational power to support the functionalities of the STS as discussed in Section 4.2.

[1] https://www.autosar.org/fileadmin/user_upload/standards/foundation/1-0/AUTOSAR_PRS_SOMEIPProtocol.pdf

[2] https://www.escrypt.com/en/news-events/autosar_security

Localised UCS+ instances are used to protect resources such as domain controllers or ECUs. However, they do *not* control the low level behaviour of such components as this imposes a safety risk. Rather, they only control the usage of high level APIs, services and functions exposed by such components. Localised UCS+ instances make usage decisions according to resource and localised context attributes as well as privileges issued by the STS. Thus, the STS handles the global context of the whole vehicle, while localised UCS+ instances manage localised authorisation contexts of individual components. To enforce the PoLP, we modified the physical architecture of UCS+ by making the PEP act as the AR of the PIP. Therefore, the PIP invokes the PEP to collect the required privileges and the PEP, in turn, requests these privileges from the wallet. Accordingly, the flow of enforcing the PoLP is shown in Figure 4 and described as follows: (1) the subject sends a request to the PEP; (2) the PEP creates an access request and invokes the CH; (3) the CH determines the required attributes to evaluate the applicable polices and invokes the corresponding PIPs; (4) if the required attribute is a privilege, the PIP asks the PEP for the attribute; (5) the PEP asks the wallet for the required privileges and the wallet presents them if they exist; (6) the PEP verifies that the privileges were issued by the STS, parses them and sends the value back to the PIP; (7) finally, the PIP returns the values to the CH, which invokes the PDP to evaluate the policy and make a decision.

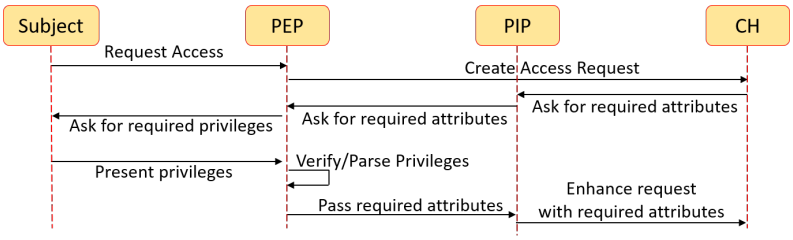

Fig. 4. Sequence diagram of enforcing the PoLP

3.4 Revocation of VCs

When an IdP revokes an external VC, such as a driving license, the IdP updates the revocation list in the VDR. The update then gets communicated to all participating entities through the VDR. If the STS is using the updated VC in an active session, then a policy re-evaluation will be triggered in that session. Based on the policy re-evaluation, the STS may revoke all internal VCs that were issued when the revoked external VC was still valid. When the STS revokes internal VCs, it also updates the VDR, thus the update gets communicated to all localised UCS+ instances. This triggers a re-evaluation in the UCS+ sessions that are using the revoked internal VC, and the corresponding access decisions will be updated according to the policy re-evaluation. We assume that OEMs define

safety procedures in the policies they install on localised UCS+ instances so that revocation of access does not cause safety threats. For instance, OEMs may define policy obligations that safely stop the vehicle before completely revoking access or delay revocation of access until the vehicle stops.

4 Experimental Evaluation

In this section, we describe two use-cases and present the outcome of the experimental evaluation of SIUV.

4.1 Use Case

One possible use case of the proposed IAM system is travelling between the borders of two countries where driving rules are different. For instance, the minimum age to drive is 17 in Denmark and 18 in Sweden. Thus, we assume a 17-year-old driver with a valid driving license in Denmark. The driver presents the license and ID credentials to the STS in the car and the STS issues a "canDrive" privilege. This privilege is then used by localised UCS+ instances to grant access to the necessary components that allow the driver to drive the car. As soon as the car crosses the borders to Sweden, the location attribute is updated and the STS re-evaluates relevant policies. In this case, the STS finds that the "canDrive" privilege is not valid anymore, so it revokes it. This is demonstrated in the non-inclusive policies of Listing 1.1. The localised UCS+ instances receive the revocation update from the VDR, and revoke access to the protected components in a safely manner. However, policies of localised UCS+ instances may include obligations that can take safety-related actions upon the revocation of privileges as shown in Listing 1.2. This use case demonstrates that SIUV supports both centralised and distributed control. Centralised control is performed by the STS that monitors the global context and verifies credentials that are relevant to all resources. Distributed control is enforced by localised UCS+ instances that monitor local contexts relevant to the protected resources only. SIUV allows the car to monitor both global and local contexts and react accordingly. Alternative solutions that only support centralised control cannot react to changes that are only relevant to a particular resource. On the other hand, solutions that only support distributed control introduce a performance overhead because global context and attributes have to be monitored by all nodes.

A car rental service is also another relevant use-case of SIUV as car owners can restrict the use of their cars according to rental agreements. For instance, a car owner can restrict the mobility of their rented car to a specific city or specific area. Thus, if the driver goes beyond the restricted area, SIUV can warn the driver or perhaps engage the autopilot to safely stop the car as defined by policies. A car owner can also use SIUV to limit the maximum speed that can be reached by the driver who rents the car. Moreover, a car owner can define a specific time period during which the car can be used by the driver according to the rental agreement. Owners can also define policies that deny access to resources that are not needed by drivers who rent the car (e.g. diagnostic interfaces).

Listing 1.1. STS policies

```
policySet privileges {
    apply permitUnlessDeny
    policy driver {
        target clause Attributes.isRegistered
        apply firstApplicable
        rule driver_dk {
            target clause Attributes.license.expiry > time.now()
                    and Attributes.license.issuer == ''borger.dk''
                    and Attributes.location == ''Denmark''
            condition Attributes.age > 17 and Attributes.ucs.step == ''ongoing''
            permit
            on permit {
                obligation canDrive { command = ''issue_privileges''
                                      canDrive = true }
        }}
        rule driver_dk_eu {
            target clause Attributes.license.expiry > time.now()
                    and Attributes.license.issuer == ''borger.dk''
            condition Attributes.age > 18 and Attributes.ucs.step == ''ongoing''
            permit
            on permit {
                obligation canDrive { command = ''issue_privileges''
                                      canDrive = true }
        }}
        rule revoke {
            deny
            on deny {
                obligation canDrive { command = ''revoke_privileges''
                                      canDrive = false }
    }}};
    policy wiperControl { ... };
    };
```

Listing 1.2. STS policies

```
policy engineControl {
    apply firstApplicable
    target clause Attributes.resourceId == ''engine''
    rule drive {
        target clause Attributes.api == ''engineAPI'' and Attributes.action == ''startEngine''
        condition Attributes.canDrive == true and Attributes.ucs.step == ''ongoing''
        permit
    }
    rule revoke {
        condition Attributes.ucs.step == ''post''
        deny
        on deny {
            obligation safeStop { command = ''autopilot''
                                  action = ''safe_stop'' }
    }}};
```

4.2 Test Cases

We implemented UCS+ and SIUV using C++ and evaluated them on a computer running Ubuntu 20.04LTS Linux OS with Core i7-9850H CPU. Modern car-ECU specifications, such as [24], include high-end processors and higher memory, which closely compares to the computational resources we use in our simulation, since UCS+ is highly optimised for similar systems as mentioned in 3.1. We

measured the time required to evaluate STS and UCS+ policies as well as the overhead cost of an increasing number of attributes. We also measured the time required to issue and verify VCs as well as the effect of the number of claims on the issuance and verification time. Finally, we evaluated the performance penalty of enforcing the PoLP especially with an increasing number of privileges. The results are shown in Figure 5 and discussed in the following subsections.

Performance of Policy Evaluation in UCS+: For this evaluation, we ran five tests with an increasing number of attributes that need to be collected by PIPs. It is necessary to note that this usually depends on the PIPs and how they collect attribute values. For instance, if a PIP needs to retrieve an attribute value over the network, then the network delay would affect the policy evaluation time. However, UCS+ caches attributes in the AT, so only policy evaluations that need an uncached attribute value would be affected by such overhead. For this reason, we only measure the time required for policy evaluation when the attributes are already cached. We ran each test 1000 rounds and observed a standard deviation of 19.2µs. Table 1 and Figure 5 show the average time required for policy evaluation in each test. The results demonstrate that our implementation is lightweight, very efficient and highly optimised, which make it suitable for embedded and safety-critical systems like vehicles.

Performance Evaluation of Issuing and Verifying VCs: We ran five different test cases with an increasing number claims. In each case, we ran the test 1000 rounds and computed the average time and the standard deviation. The observed standard deviation was 13.8µs and the results are shown in Table 2. The results show that the number of claims does not have a significant effect on the efficiency of issuing and verifying VCs. In addition, they show that the time required to issue or verify a typical VC is less than 0.5ms. This is because we used Ed25519 digital signatures, which are faster than any ECDSA or EdDSA instances in terms of signature generation and verification.

Performance Evaluation of the PoLP Enforcement: The implementation of the PoLP includes back and forth communication between the wallet and the PEP, and between the PEP and the PIP. Undoubtedly, this introduces a performance overhead especially because the PIP asks for privileges sequentially and synchronously. To measure this overhead, we conducted 4 test cases with an increasing number of privileges. The average time to enforce the PoLP was computed as well as the standard deviation after running each test case 1000 times. We noted a standard deviation of 44.7µs. The results presented in Table 3 show that the overhead increases significantly and linearly as the number of required privileges increases. However, the number of privileges required for a specific action is not expected to be high, so such overhead can be tolerated. Nonetheless, we plan to explore more efficient approaches to enforce the PoLP.

Table 1. Average time to evaluate ALFA policies as the number of attributes increases

Number of attributes	**1**	**3**	**5**	**10**	**20**
Average time to evaluate a policy (in µs)	9	11	18	27	29

Table 2. Average time to issue and verify a VC as function of the number of claims

No. of claims → / **Avg. time (µs)** ↓	**1**	**3**	**5**	**10**	**20**
Issue a VC (µs)	86	97	105	136	196
Verify a VC (µs)	115	119	125	142	172

Table 3. Average time to enforce the PoLP as function of privilege number

No. of privileges → / **Avg. time (µs)** ↓	**1**	**3**	**5**	**10**
Enforce the PoLP	300	952	1554	3136

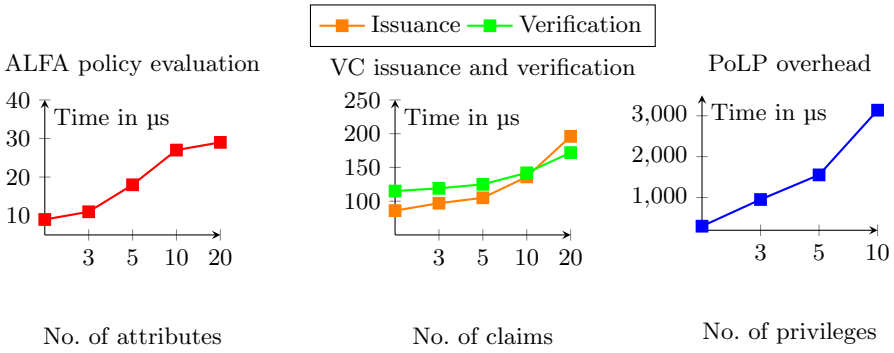

Fig. 5. Performance results of issuing and verifying VCs, and enforcing the PoLP

5 Conclusion and Future Work

We demonstrated the usage of VCs and UCON to support dynamic and stateful identity management and continuous usage control in smart vehicles. We specifically introduced a highly efficient implementation of the UCON model designated as UCS+. We also introduced a centralised STS that exchanges identity VCs with privilege VCs that determine what subjects are allowed to do inside the vehicle. We also proposed an SOA-based architecture for a vehicular IAM that uses a centralised STS for identity management and distributed UCS+ instances to protect resources. SIUV addresses the problem of dynamic identity management as well as statefull access control in smart vehicles whose contexts change continuously. The evaluation of the proposed system showed promising results and significant improvements over similar works.

Future directions should aim at providing anonymity or pseudonymity and maximizing privacy of the subject such that issuers or verifiers cannot identify the subject properties. Particularly, we plan to investigate the addition of a negotiation protocol between the wallet and other components in order to regulate the disclosure of credentials and privileges. In addition, we intend to explore selective disclosure and zero knowledge schemes as well as predicate proofs. This allows to minimise identity information sharing while still being able to create a

fully functional ecosystem of multi-party credential exchange and identity verification. We also intend to use JSON-LD, to interpret and serialize linked data in JSON and be able to support signature sets, signature chaining etc.

References

1. Ammar, M., Janjua, H., Thangarajan, A., Crispo, B., Hughes, D.: Securing the On-board Diagnostics Port (OBD-II) in Vehicles. 8th Embedded Security in Cars (ESCAR USA) (2020)
2. AUTOSAR: Explanation of Adaptive Platform Design (March 2019), https://www.autosar.org/fileadmin/user_upload/standards/adaptive/19-11/AUTOSAR_EXP_PlatformDesign.pdf
3. Bernardini, C., Asghar, M.R., Crispo, B.: Security and privacy in vehicular communications: Challenges and opportunities. Vehicular Communications **10**, 13–28 (2017)
4. Bernstein, D.J., Duif, N., Lange, T., Schwabe, P., Yang, B.Y.: High-speed high-security signatures. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 124–142. Springer (2011)
5. Brendel, J., Cremers, C., Jackson, D., Zhao, M.: The provable security of ed25519: theory and practice. IEEE Security & Privacy (2020)
6. Burkacky, O., Deichmann, J., Doll, G., Knochenhauer, C.: Rethinking car software and electronics architecture (February 2018), https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/rethinking-car-software-and-electronics-architecture
7. Burkacky, O., Deichmann, J., Klein, B., Pototzky, K., Scherf, G.: Cybersecurity in automotive: Mastering the challenge (June 2020), https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/cybersecurity-in-automotive-mastering-the-challenge
8. Chen, L., Moody, D., Regenscheid, A., Randall, K.: Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters. Tech. rep., National Institute of Standards and Technology (2019)
9. Deichmann, J., Klein, B., Scherf, G., Rupert, S.: The race for cybersecurity: Protecting the connected car in the era of new regulation (October 2019), https://mckinsey.com/industries/automotive-and-assembly/our-insights/the-race-for-cybersecurity-protecting-the-connected-car-in-the-era-of-new-regulation
10. Denis, F.: libsodium: A modern and easy-to-use crypto library (2017), https://libsodium.gitbook.io/doc/
11. Dimitrakos, T., Dilshener, T., Kravtsov, A., La Marra, A., Martinelli, F., Rizos, A., Rosetti, A., Saracino, A.: Trust Aware Continuous Authorization for Zero Trust in Consumer Internet of Things. In: 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). pp. 1801–1812 (2020). https://doi.org/10.1109/TrustCom50675.2020.00247
12. Dürrwang, J., Braun, J., Rumez, M., Kriesten, R.: Security evaluation of an airbag-ECU by reusing threat modeling artefacts. In: 2017 International Conference on Computational Science and Computational Intelligence (CSCI). pp. 37–43. IEEE (2017)
13. Hamad, M., Prevelakis, V.: Secure APIs for Applications in Microkernel-based Systems. In: ICISSP. pp. 553–558 (2017)

14. Hu, V.C., Ferraiolo, D., Kuhn, R., Friedman, A.R., Lang, A.J., Cogdell, M.M., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K., et al.: Guide to attribute based access control (abac) definition and considerations (draft). NIST special publication **800**(162) (2013)
15. Josefsson, S., Liusvaara, I.: Rfc8032: Edwards-curve digital signature algorithm (eddsa). Request for Comments, IETF (2017)
16. Kim, D.K., Song, E., Yu, H.: Introducing Attribute-Based Access Control to AUTOSAR. Tech. rep., SAE Technical Paper (2016)
17. Lazouski, A., Martinelli, F., Mori, P.: A prototype for enforcing usage control policies based on XACML. In: International Conference on Trust, Privacy and Security in Digital Business. pp. 79–92. Springer (2012)
18. Miller, C., Valasek, C.: Remote exploitation of an unaltered passenger vehicle. Black Hat USA **2015**, 91 (2015)
19. OASIS: Abbreviated language for authorization Version 1.0 (2015), https://bit.ly/2UP6Jza
20. OASIS: eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01 (2017), http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-en.html
21. Park, J., Sandhu, R.: The UCONABC usage control model. ACM Transactions on Information and System Security (TISSEC) **7**(1), 128–174 (2004)
22. Rumez, M., Duda, A., Gründer, P., Kriesten, R., Sax, E.: Integration of attribute-based access control into automotive architectures. In: 2019 IEEE Intelligent Vehicles Symposium (IV). pp. 1916–1922. IEEE (2019)
23. Rumez, M., Grimm, D., Kriesten, R., Sax, E.: An Overview of Automotive Service-Oriented Architectures and Implications for Security Countermeasures. IEEE Access **8**, 221852–221870 (2020)
24. Samsung: Automotive Processor Exynos Auto V9, https://www.samsung.com/semiconductor/minisite/exynos/products/automotiveprocessor/exynos-auto-v9/
25. Sporny, M., Longley, D., Chadwick, D.: Verifiable credentials data model 1.0. Tech. rep., W3C (November 2019), https://www.w3.org/TR/vc-data-model/
26. Wouters, L., Marin, E., Ashur, T., Gierlichs, B., Preneel, B.: Fast, furious and insecure: Passive keyless entry and start systems in modern supercars. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 66–85 (2019)