# Security and Trust in Safety Critical Infrastructures

Zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.)
Genehmigte Dissertation von Maria Zhdanova aus Moskau, Russland
Tag der Einreichung: 31.12.2021, Tag der Prüfung: 25.05.2022

1. Gutachten: Prof. Dr. Michael Waidner
2. Gutachten: Univ.-Prof. Dipl.-Ing. Mag. Dr. techn. Edgar Weippl
3. Gutachten: Prof. Dr. Christoph Krauß
Darmstadt 2022

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Computer Science
Department

Security in Information
Technology

Fraunhofer Institute for
Secure Information
Technology SIT

Fraunhofer
SIT

Security and Trust in Safety Critical Infrastructures

Accepted doctoral thesis by Maria Zhdanova

1. Review: Prof. Dr. Michael Waidner
2. Review: Univ.-Prof. Dipl.-Ing. Mag. Dr. techn. Edgar Weippl
3. Review: Prof. Dr. Christoph Krauß

Date of submission: 31.12.2021
Date of thesis defense: 25.05.2022

Darmstadt, Technische Universität Darmstadt

Jahr der Veröffentlichung der Dissertation auf TUprints: 2022

For Family

# Erklärungen laut Promotionsordnung

## §8 Abs. 1 lit. c PromO

Ich versichere hiermit, dass die elektronische Version meiner Dissertation mit der schriftlichen Version übereinstimmt.

## §8 Abs. 1 lit. d PromO

Ich versichere hiermit, dass zu einem vorherigen Zeitpunkt noch keine Promotion versucht wurde. In diesem Fall sind nähere Angaben über Zeitpunkt, Hochschule, Dissertationsthema und Ergebnis dieses Versuchs mitzuteilen.

## §9 Abs. 1 PromO

Ich versichere hiermit, dass die vorliegende Dissertation selbstständig und nur unter Verwendung der angegebenen Quellen verfasst wurde.

## §9 Abs. 2 PromO

Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 31.12.2021

Maria Zhdanova

# Zusammenfassung

Kritische Infrastrukturen wie Straßenfahrzeuge und Eisenbahnen durchlaufen einen tiefgreifenden Wandel, der die Abhängigkeit derer Steuerung und Betrieb von der Informationstechnologie erhöht. Durch die zunehmende Vernetzung dieser sicherheitskritischen Systeme entstehen neue komplexe Kommunikationsinfrastrukturen, die gleichzeitig dem effizienten Management der Betriebsprozesse dienen und den Informationsaustausch mit verschiedenen Dritten ermöglichen. Mit der Einführung des digitalen Stellwerks wird die Bahn-Leit- und Sicherungstechnik (LST) zum "Internet der Bahn-Dinge", in dem sicherheitskritische Komponenten auf Mehrzweckplattformen realisiert und über IP-basierte Standardnetze verbunden sind. Ebenso führt die massive Verbreitung von Elektrofahrzeugen und deren Energiebedarf zu einer neuen Vehicle-to-Grid (V2G) Lade-Infrastruktur, in der Fahrzeuge mit Stromnetzen und Dienstanbietern verbunden sind, um Lade- und Entladevorgänge zu koordinieren und zur Netzstabilität beizutragen. So erlaubt die Plug-and-Charge-Funktion aus der V2G-Kommunikationsnorm ISO 15118 einem Elektrofahrzeug weitgehend automatisiert, mithilfe eigener Identitätsmerkmale, auf Lade- und Mehrwertdienste zuzugreifen, Ladepläne dynamisch auszuhandeln und das Stromnetz als verteilte Energieressource zu unterstützen.

Die rasche Einführung dieser fortschrittlichen Technologien wird durch wirtschaftliche und politische Entscheidungen wie den EU Green Deal für Klimaneutralität vorangetrieben. Aufgrund der komplexen Anforderungen und langen Standardisierungs- und Entwicklungszyklen geraten dabei auch Normen und Vorschriften unter Druck, die traditionell eine Schlüsselrolle beim Betrieb und Schutz kritischer Infrastrukturen spielen.

In dieser Dissertation untersuchen wir zukünftige V2G- und Eisenbahn-LST-Systeme in Bezug auf sicherer Kommunikation, Plattformsicherheit sowie des Zusammenwirkens von Security und Safety. Eines der wichtigsten Ziele dabei ist es, die vorgeschlagenen Sicherheitslösungen auch in kommenden branchenspezifischen Standards zu etablieren und so die korrekte Umsetzung in realen Produkten zu ermöglichen.

Wir untersuchen zunächst die Sicherheit von V2G-Kommunikationsprotokollen und die Anforderungen für die sichere Bereitstellung von Diensten über Ladekommunikation. Wir schlagen ein neues Plug-and-Patch-Protokoll vor, das die sichere Aktualisierung von Elektrofahrzeugen als Mehrwertdienst in den Ladevorgang integriert. Da Fahrzeuge auch am Energiehandel teilnehmen können, indem sie zuvor gespeicherte Energie in das Stromnetz, ins Heim oder in andere Fahrzeuge einspeisen, untersuchen wir anschließend Betrugserkennungsmethoden, die zur Identifikation von Fehlverhalten und Manipulation benutzt werden können.

Um eine zuverlässige Grundlage für die sichere V2G-Kommunikation zu schaffen, entwickeln wir drei Sicherheitsarchitekturen für V2G-Systeme, die auf Basis von Hardware-Sicherheitsankern die gegenseitige Vertrauensetablierung in der V2G-Kommunikation ermöglichen. Wir integrieren diese Architekturen in V2G-Protokolle für Lastmanagement, E-Mobilitäts- und Mehrwertdienste und analysieren die jeweils erreichte Leistung und Sicherheit.

Der letzte Aspekt dieser Arbeit ist die Ko-Entwicklung von Security und Safety Prozessen, die für den wirksamen Schutz vernetzter sicherheitskritischer Systeme unerlässlich ist. Wir untersuchen für zwei Anwendungsszenarien, Batterieladesystem im Fahrzeug (EVCS) und Objekt Controller im Bahn-LST-System, wie Trusted Computing Sicherheitskonzepte angewandt werden können um sowohl die erforderlichen Security- als auch die Safety-Eigenschaften zu gewährleisten. Im Fall von EVCS setzen wir die Vertrauensgrenzen für die Safety-Funktionalitäten (zertifizierte Konfiguration) in Beziehung zu den Security-Vertrauensgrenzen und entwerfen eine neue Sicherheitsarchitektur, die Safety-Eigenschaften mithilfe von Security-Assertions durchsetzt. Im Anwendungsfall Eisenbahn konzentrieren wir uns auf die Rückwirkungsfreiheit (Trennung) zwischen den Domänen und entwickeln eine Sicherheitsarchitektur, die sichere Koexistenz von Anwendungen mit verschiedener Kritikalität auf derselben Hardware-Plattform ermöglicht.

Unsere Lösungen wurden dem Gremium ISO/TC 22/SC 31/JWG 1 für die Normenreihe ISO 15118 und dem DKE AK "Informationssicherheit für Elektromobilität", der für die dazugehörigen Anwendungsregeln zuständich ist, vorgestellt. Unsere Sicherheitserweiterung ist in der neuesten Ausgabe ISO 15118-20 vom April 2022 integriert. Mehrere Hersteller haben bereits mit der Konzeptvalidierung ihrer zukünftigen Produkte unter Verwendung unserer Ergebnisse begonnen.

Die vorgestellten Konzepte und Analysen tragen damit fundamental zur Verbesserung der Sicherheit von E-Mobilitäts- und Bahnanwendungen bei und erhöhen die Widerstandsfähigkeit dieser sicherheitskritischen Infrastrukturen.

# Abstract

Critical infrastructures such as road vehicles and railways are undergoing a major change, which increases the dependency of their operation and control on Information Technology (IT) and makes them more vulnerable to malicious intent. New complex communication infrastructures emerge using the increased connectivity of these safety-critical systems to enable efficient management of operational processes, service provisioning, and information exchange for various (third-party) actors. Railway Command and Control Systems (CCSs) turn with the introduction of digital interlocking into an "Internet of Railway Things", where safety-critical railway signaling components are deployed on common-purpose platforms and connected via standard IP-based networks. Similarly, the mass adoption of Electric Vehicles (EVs) and the need to supply their batteries with energy for charging has given rise to a Vehicle-to-Grid (V2G) infrastructure, which connects vehicles to power grids and multiple service providers to coordinate charging and discharging processes and maintain grid stability under varying power demands. The Plug-and-Charge feature brought in by the V2G communication standard ISO 15118 allows an EV to access charging and value-added services, negotiate charging schedules, and support the grid as a distributed energy resource in a largely automated way, by leveraging identity credentials installed in the vehicle for authentication and payment.

The fast deployment of this advanced functionality is driven by economical and political decisions including the EU Green Deal for climate neutrality. Due to the complex requirements and long standardization and development cycles, the standards and regulations, which play the key role in operating and protecting critical infrastructures, are under pressure to enable the timely and cost-effective adoption.

In this thesis, we investigate security and safety of future V2G and railway control and command systems with respect to secure communication, platform assurance as well as safety and security co-engineering. One of the major goals in this context is the continuous collaboration and establishment of the proposed security solutions in upcoming domain-specific standards, thus ensuring their practical applicability and prompt implementation in real-world products.

We first analyze the security of V2G communication protocols and requirements for secure service provisioning via charging connections. We propose a new Plug-and-Patch protocol that enables secure update of EVs as a value-added service integrated into the V2G charging loop. Since EVs can also participate in energy trading by storing and feeding previously stored energy to grid, home, or other vehicles, we then investigate fraud detection methods that can be employed to identify manipulations and misbehaving users.

In order to provide a strong security foundation for V2G communications, we propose and analyze three security architectures employing a hardware trust anchor to enable trust establishment in V2G communications. We integrate these architectures into standard V2G protocols for load management, e-mobility services and value-added services in the V2G infrastructure, and evaluate the associated performance and security trade-offs.

The final aspect of this work is safety and security co-engineering, i.e., integration of safety and security processes vital for the adequate protection of connected safety-critical systems. We consider two application scenarios, Electric Vehicle Charging System (EVCS) and Object Controller (OC) in railway CCS, and investigate how security methods like trusted computing can be applied to provide both required safety and security properties. In the case of EVCS, we bind the trust boundary for safety functionality (certified configuration) to the trust boundary in the security domain and design a new security architecture that enforces safety properties via security assertions. For the railway use case, we focus on ensuring non-interference (separation) between these two domains and develop a security architecture that allows secure co-existence of applications with different criticality on the same hardware platform.

The proposed solutions have been presented to the committee ISO/TC 22/SC 31/JWG 1 that develops the ISO 15118 standard series and to the DKE working group "Informationssicherheit für Elektromobilität" responsible for the respective application guidelines. Our security extension has been integrated in the newest edition ISO 15118-20 released in April 2022. Several manufacturers have already started concept validation for their future products using our results.

In this way, the presented analyses and techniques are fundamental contributions in improving the state of security for e-mobility and railway applications, and the overall resilience of safety-critical infrastructures to malicious attacks.

# Acknowledgments

I would like to thank everyone who supported me during this long and challenging journey and helped me to fulfill my dream and successfully finish my PhD.

First, I would like to thank my advisor Prof. Dr. Michael Waidner for giving me this opportunity and freedom to pursue my research at the Fraunhofer Institute for Secure Information Technology (SIT) and for providing his guidance during my PhD. I am especially thankful for the nomination for the TALENTA Program of the Fraunhofer Society, which had a major impact on my doctorate as well as my professional and personal development.

I am also very grateful to Prof. Dr. techn. Edgar Weippl and Prof. Dr. Christoph Krauß who agreed to be my reviewers and provided the detailed feedback on this thesis. Moreover, I would like to express my gratitude to my examination committee members Prof. Dr. Marc Fischlin, Prof. Dr. Dr. eh. Dieter Fellner, and Prof. Sebastian Faust, Ph.D. Thank you very much for the interesting questions during my defense.

I would like to thank my present and former colleagues and students at Fraunhofer SIT and TU Darmstadt who contributed to an open and supportive research environment by working together on inventive and inspiring projects, and openly exchanging thoughts, ideas, and scientific arguments. My special thanks go to my co-authors Dr. Roland Rieke and Jürgen Repp who introduced me to the model-based process analysis and the Hessian culture and to Markus Springer, Daniel Zelle, Lukas Buschlinger, and Dustin Kern with whom we explored the new and fascinating field of e-mobility and built awesome demonstrators.

I was also deeply honored to collaborate in research projects with outstanding experts from external organizations in academia, industry, and standardization and to create new secure technologies and product designs with them. I want particularly to thank Dr. Martin Simons and Alexander Irrgang, colleagues from the ISO 15118 working group, AG CYSIS and the HASELNUSS project team.

I would like to thank my TALENTA mentors and co-mentees for their encouragement, insightful advice, and experience sharing, which helped my successful graduation a lot.

I thank everyone who assisted me in organizing my defense day and everyone who shared this special day with me.

Last but not least, I would like to thank my family for their love and unlimited support, for them always believing in me and being there for me during my PhD and beyond. This thesis would not have been possible without my parents Sergey and Natalia, who raised me to value education and knowledge most and always did everything possible for me and my sister to fully develop our potential. Thank you for teaching me what kind of researcher I want to be. I thank my little sister Tatiana for sharing her confidence and wisdom and continuously raising the bar for me. The biggest thank goes to my husband Steffen for his patience, courage, and optimism, for advising me and coaching me all this time, and for helping me get my PhD done. I am very lucky to have you.

*Thank you!*

# Contents

# Part I.

# Research Problem and Background

# 1. Introduction

## 1.1. Motivation

Critical transport infrastructures such as road vehicles and railways are undergoing a major transformation. In the case of the road transport, this transformation concerns the expansion of e-mobility, i.e., the ongoing replacement of traditional vehicles with EVs powered by a rechargeable battery. E-mobility is considered one of the main technologies for achieving the Paris Agreement objectives, therefore, this development is supported by sustainability policies and financial incentives in many countries. Consequently, the worldwide adoption of EVs including fully battery-powered and plug-in hybrid vehicles is growing, with the 10 million mark passed in 2020 [35]. Since the need to charge EV batteries causes an extra load on power grids, which need to supply vehicles with electricity on demand, this change also effects the energy system. In order to enable coordinated EV charging and energy billing, a new complex communication infrastructure emerged, which connects backend systems of service providers, such as Distribution System Operators (DSOs), Charge Point Operators (CPOs), and E-Mobility Service Providers (eMSPs) with EVs and Charge Points at the front end. This includes the introduction of the V2G technology that allows EVs to communicate with the power grid and service providers to negotiate optimal conditions for charging and discharging and mitigate peak power demands (i.e., smart charging). To provide additional grid support during peak times or power supply failures, the storage capacity of EV batteries can be used as a flexible energy resource to handle power fluctuations [36].

The international standard ISO 15118 [37, 38, 39] defines the V2G communication protocol using High Level Communications (HLCs) to coordinate the bidirectional power transfer between EVs and Charge Points. Beside enabling the V2G functionality, this standard also largely automates the actions necessary on the side of drivers to charge their vehicles by introducing the Plug-and-Charge (PnC) mechanism (or Park-and-Charge, in case of wireless power transfer). In particular, the standard automates authentication of EVs, authorization of charging sessions, and collection of billing information by providing each vehicle with identity credentials created by its manufacturer and/or eMSP and based on the V2G Public Key Infrastructure (PKI). Since ISO 15118 describes only the interface between the vehicle and the charger, further protocols that connect the charger to backend systems of the service providers are developed [40].

As the V2G infrastructure implements safety-critical processes such as EV charging, load management, or demand response by connecting safety-critical high-wattage systems, malicious attacks on this infrastructure may disrupt energy supply and cause safety hazards [41]. For example, an adversary can damage the vehicle's battery and even make it catch fire by manipulating the battery's controller [42, 43]. Moreover, if adversaries manage to bring multiple EVs under their control, such a botnet of high-wattage devices can be used to cause blackouts in local energy grids [44, 45, 46]. If V2G communications are not protected, adversaries can eavesdrop on charging sessions [47, 48] and under certain circumstances even inject manipulated messages [49] to disrupt EV charging. Since the PnC credentials enable access to V2G services and provide the basis for billing, they are an attractive target for adversaries and must be protected against illegitimate access and misuse [50, 51, 52]. These credentials are at least temporary stored in the backend systems before being installed into the vehicle or encrypted for the delivery. Thus, a successful attack on the corporate network may result in the compromise of these credentials [53, 54, 55]. EVs and Charge Points are also vulnerable to malicious attacks. Both local and remote attacks on chargers from multiple manufacturers have been recently demonstrated [56, 57, 58]. Similarly, car hacking research [59, 60, 61, 62] shows that an adversary can get access to virtually any Electronic Control Unit (ECU) in a modern vehicle, even remotely.

The safety-critical railway infrastructure is undergoing the digitalization process characterized by the introduction of connected IT-based systems into the previously analogue rail operation processes and thus expected to face similar risks. One of the first examples is the ransomware WannaCry, which hit the systems of the Deutsche Bahn (DB) in Germany in 2017. Railway CCSs responsible for the safe operation of trains are also changing through the adoption of common-purpose platforms and IP-based communication networks and the increased connectivity between all layers of operations. For instance, the DB plans to digitalize its infrastructure by 2037 [63] as part of the NeuPro project starting with the digitalization of OCs located at the tracks to control field elements. This added connectivity may open railway CCSs to large-scale attacks, where an adversary can instantaneously seize control over multiple rail segments and trigger hazardous situations with potentially devastating consequences. However, the mandatory safety certification of these systems does not consider protections against malicious actions, neither co-engineering approaches for safety and security are in place. Moreover, integrating security functionalities into a safety-certified systems without voiding this certification is a major challenge [64].

The purpose of this thesis is to investigate security and safety of V2G and railway command and control systems with respect to secure communications, platform assurance as well as safety and security co-engineering. One of the major goals in this context is the continuous collaboration and the establishment of the proposed security solutions within relevant domain-specific standards to ensure their practical applicability and timely adoption in real-world products.

## 1.2. Outline and Summary of Contributions

Following this introduction, Chapter 2 provides the relevant background to the V2G ecosystem with the focus on e-mobility applications and protocols. We also provide some basics on the security functionalities specified in the open standard Trusted Platform Module (TPM) 2.0 used as building blocks for the proposed solutions.

In this work, we make contributions concerning two major areas. The V2G communication security aspects are covered in Part II, while safety and security co-engineering and platform assurance based on an Hardware Security Module (HSM) acting as trust anchor is addressed in Part III. Concluding summary and directions for future research are provided in Part IV closing this thesis.

Both Part II and Part III contain partially revised versions of selected peer-reviewed publications by the author, where the author provided major contributions to the presented analyses and concepts, although the respective referenced publications often use an alphabetical or alphabetical-by-institute listing of authors.

The contributions of this thesis are transferred to the relevant standardization committees including the Security Task Force of ISO/TC 22/SC 31/JWG 1 that develops the ISO 15118 standard series and to DKE AK "Informationssicherheit für Elektromobilität" responsible for the respective application guidelines. Our new security extension to ISO 15118 has been integrated in the ISO/FDIS 15118-20 with the planned issue date of the international standard in Q2 2022. In particular, we contribute to the following areas with this work:

1. We present a new secure service protocol Plug-and-Patch (PnP) for remote vehicle updates leveraging the connectivity provided by the V2G infrastructure for the service provisioning as an alternative to the Over-the-Air (OTA) updates delivered via the proprietary telematics link.

2. We propose the security architectures TrustEV, HIP and HIP 2.0 that protect vehicle identities used in V2G scenarios and support different provisioning and enrollment use cases. Due to cryptographically secure binding of the corresponding credentials to the vehicle, they can be employed to establish trust in load management, e-mobility services and value-added services. Currently, the trust relationship builds solely on the usage of a dedicated V2G PKI with unrealistic security assumptions. This thesis addresses this gap by presenting HSM-based solutions and by integrating them into the V2G protocols in a standard-conform and backward compatible way.

3. We investigate security and safety co-engineering for safety-critical Electric Vehicle Charging System and design a new security architecture `secEVCS` enabling the secure binding between components comprising the safe configuration. This binding protects the V2G infrastructure against vehicles with manipulated or counterfeit charging systems and reduces the risk of safety-related security threats.

4. We further investigate security and safety co-engineering for the safety-critical railway CCS and develop a security architecture Haselnuss Reference Architecture (HRA) that enables the secure co-existence of applications with different criticality on the same hardware platform. By assuring the freedom of interference, HRA adds the necessary security functionality to the safety-critical system and preserves the safety certification.

5. We analyze potential fraud scenarios applicable to V2G services and devise a new FCD method for fraud detection in distributed payment systems based on the event-driven process modeling approach.

### 1.2.1. Securing Vehicle-to-Grid Communications

The Vehicle-to-Grid (V2G) infrastructure supports a variety of applications, including the management of processes related to energy supply, V2G session handling and collection of billing information, as well as various additional services using the V2G connectivity. In Part II, we investigate different aspects of the V2G communication security. We analyze the security of the established V2G communication protocols to identify potential gaps and design a secure protocol for an example value added service. We devise a method to monitor the behavior of service users and detect action patterns considered potentially malicious.

### Securing Value-Added Services for Electric Vehicle Charging

In the recent years, multiple communication protocols have been developed to enable the information exchange and service provisioning in V2G infrastructures. The most prominent examples are the ISO 15118 standard series, which define the protocol stack for the communication between Electric Vehicle (EV) and Charge Point, and Open Charge Point Protocol (OCPP), which further connects Charge Point to backend systems to enable service management and billing. The option to render services to drivers while they have to wait for the EV battery to recharge is expected to motivate the industry participation in the development of the charging infrastructure. However, the capabilities of the V2G infrastructure with respect to secure service provisioning have not been analyzed. ISO 15118-2 acknowledges this option but only provides a generic service discovery interface, whereas the message flow necessary to integrate Value Added Services (VAS) into the V2G charging loop is not considered.

Chapter 3 provides an extensive analysis of VAS for EV charging and investigates the security of V2G communication protocols with regard to secure service provisioning. Based on these results, we propose a new protocol Plug-and-Patch (PnP) that enables secure remote updates of EVs by their Original Equipment Manufacturers (OEMs) via the existing V2G channel (i.e., powerline for cable-based charging or wireless for inductive charging) as an additional service offered by charge points. Our protocol protects updates against untrusted intermediaries and powerful adversaries and allows the OEM to verify whether the vehicle received the update using TPM-based attestation, for operational and liability purposes. We provide an integration concept for our exemplary service protocol into ISO 15118-2 and OCCP 2.0 and evaluate the associated security and performance trade-offs using a Proof-of-Concept (PoC). One of our main conclusions is that the proposed approach is well suited to distribute patches for smaller Electronic Control Units (ECUs), without car recalls or update provider lock-ins. However, larger updates may need to be performed in an incremental fashion, due to potentially limited bandwidth of the charging connection. These results apply to any arbitrary VAS.

*This contribution is based on a peer-reviewed paper [3] published in collaboration with Lucas Buschlinger and Markus Springer. The author made significant contributions to the problem definition, security analysis, protocol design, and planing of the evaluation, as well as management and writing of the paper.*

### Fraud Detection in Distributed Payment Systems

In order to prevent financial fraud related to the service consumption and avoid potential money and reputation loss, service providers strongly depend on fraud detection systems, whose usage is also often mandated by anti-fraud regulations. For example, due to Anti-Money Laundering (AML) regulations valid in most countries, it is compulsory for service providers to report Money Laundering (ML) activities. Thus, reliable ML detection is crucial. The common approach to fraud detection is to use classical statistical methods including machine learning and data mining. However, these methods rely on training data, which can be difficult to obtain for fraudulent transactions. Moreover, they often produce results that are hard to interpret and may have difficulties to distinguish fraud from the legitimate user behavior, if those have close characteristics. Since billing of customers for the usage of the charging service or VAS belongs to the tasks realized in the V2G infrastructure, this problem is also relevant for E-Mobility Service Providers (eMSPs) and Contract Clearing House (CCH) responsible for this task.

Chapter 4 investigates fraud detection methods that can be employed to identify misbehaving EVs, too. As an exemplary fraud scenario, we consider micro-structuring of funds (smurfing) in the Mobile Money Transfer (MMT). We propose a new method for fraud detection that can identify complete fraud chains participating in ML with high recognition performance. In contrast to the classical detection schemes, our FCD method builds on a model-based approach for event-driven process analysis, which requires only process specifications as input and does not depend on any prior information about fraud patterns. To evaluate the proposed solution, a comparative study with several machine learning algorithms has been performed using synthetic data produced by an MMT simulator from [65]. In these experiments, FCD achieved better recognition performance than the standard techniques. Moreover, the usage of process models allowed for the easy interpretation of alerts helping analysts to report fraud more efficiently. The only potential bottleneck of FCD is the computational performance if the number of simultaneously monitored user processes is very high. In this case, methods like time sliding windows can be applied for the performance optimization. Therefore, we argue that our new method provides a valuable extension to fraud detection systems, which can also be deployed in parallel with the classical schemes, for example, only for suspect accounts.

*This contribution is based on a peer-reviewed paper [5] nominated for the best paper award at the International Conference on Availability, Reliability and Security (ARES). This is a collaborative work with the colleagues from the EU project MASSIF, where the author made major contributions to the problem definition, requirements analysis, fraud detection method, and analysis of experimental results, as well as management and writing of the paper.*

## 1.2.2. Platform Assurance in Safety-Critical Infrastructures

In Part III, we examine security risks in the V2G systems and railway Command and Control Systems (CCSs) that can have both security and safety implications. We design security architectures to assure EV identities under various provisioning strategies, which comply with the domain-specific standards and thus can be easily integrated into existing solutions. We investigate safety and security co-engineering and develop security designs that address the following aspects: the binding between safety and security trust boundaries to assure safe operation, and the freedom of interference between security and safety to be able to add security functions to safety-certified systems without voiding their certification.

### Enabling Trust in V2G Communications

In the context of ISO 15118, the V2G authentication is based on the dedicated V2G Public Key Infrastructure (PKI). Thereby, the vehicle's identity credentials are created by the OEM and its eMSPs that offer the contract-based charging service Plug-and-Charge (PnC). Since these credentials are also used for billing purposes, they can be qualified as payment credentials and thus the conformance to the Payment Card Industry Data Security Standard (PCI DSS) may be mandatory. Moreover, as these credentials enable access to the safety-critical infrastructure, the highest Level of Assurance (LoA) (cf. ISO/IEC 29115) need to be achieved, i.e., a certified Hardware Security Module (HSM) needs to be integrated for secure storage of private keys and cryptographic operations on the vehicle. Recent vehicle attacks by Miller and Valasek as well as well-known backend attacks like Sony hack or Certificate Authority (CA) breaches prove that the associated security risks of credential compromise are real. However, neither the standard ISO 15118 nor the respective application guideline VDE-AR 2802 consider the security of V2G credentials during storage and usage in the vehicle as well as assurance that these credentials can only be used by the vehicle, for which they were issued.

Chapter 5 investigates secure identities for electric vehicles. We consider different credential provisioning and enrollment strategies supported by the standard and/or application guideline and introduce security architectures TrustEV, HIP and HIP 2.0 that address each particular scenario. The proposed security functionality assumes a certified automotive-class HSM solutions to be available in the vehicle. TrustEV provides secure provisioning, storage, and usage of PnC credentials in an EV equipped with a HSM. HIP extends this approach to secure and verifiable generation of all cryptographic keys in the vehicle's HSM followed by the credential enrollment at an eMSP, whereby the eMSP receives an assurance that the customer's vehicle can use the contract credential only if they were generated and stored securely. HIP 2.0 improves this solution by supporting established CA processes such as using Certificate Signing Requests (CSRs), out-of-band credential provisioning via channels outside of the ISO 15118 communication according the application guideline VDE-AR 2802, and the revised certificate update process of the upcoming edition ISO 15118-20. TrustEV aims to protect against car hackers, while HIP and HIP 2.0 also address backend hackers by outsourcing key generation to a secure environment on the vehicle. To instantiate our design, we employ Trusted Platform Module (TPM) 2.0 as an automotive HSM. Since TPM is an open standard, with open source software stack and hardware easily available, this decision provides for reproducible benchmarks for PoCs and makes the integration into an international standard easier due to the well specified formats and Application Programming Interfaces (APIs). Therefore, as part of our contribution, we provide concepts for integrating the TPM 2.0 credential protection feature, direct key import and enhanced authorization into the processes of ISO 15118 and VDE-AR 2802 assuring backward compatibility. We analyze and discuss the security properties of the proposed designs considering strong adversaries and evaluate several performance aspects to check the compatibility of the solution with the relevant constraints of ISO 15118 using a proof-of-concept implementation.

One of the main conclusions is that it is challenging to provide the required protection level and remain conform to the ISO 15118 standard. Since additional security operations incur computational and communication overhead, which may not be accounted for in the standard, the straightforward integration may be problematic. Therefore, our results provide important insights about potential overheads of the security assurance that should be considered in the future standardization efforts.

*This contribution is based on peer-reviewed papers [1], [7], and [8] published in collaboration with Dustin Kern, who performed most of the implementation and performance evaluation, Christoph Krauß and Andreas Fuchs, who provided expertise on the TPM 2.0 specification and the open-source software stack. The author made a major contribution to the problem definition, definition of security and functional requirements, architectural design, and planing of the evaluation, as well as management and writing of the papers. The author has also transferred the proposed solutions to the ISO 15118 standardization committee and successfully integrated TrustEV into the upcoming ISO/FDIS 15118-20 edition of the standard. HIP and HIP 2.0 were also considered but their integration was postponed till the next revision round. Several OEMs has already started the concept validation for their future products using our results.*

### Securing Safety-critical Electric Vehicle Charging Systems

Since the V2G infrastructure implements safety-critical processes such as EV charging, load management, or demand response by connecting safety-critical high-wattage systems, malicious attacks on this infrastructure may disrupt energy supply and cause safety hazards. For example, an adversary can damage the vehicle's battery and even make it catch fire by manipulating the battery's controller. Moreover, if adversaries manage to bring multiple EVs under their control, such a botnet of high-wattage devices can be used to cause blackouts in local energy grids. However, only one domain-specific standard SAE J3061 considers an integrated safety and security lifecycle for automotive systems. Moreover, our background research did not reveal solutions that address these particular problems from both points of view.

Chapter 6 investigates safety and security co-engineering aspects for V2G systems, with the focus on the Electric Vehicle Charging System (EVCS) comprising the Electric Vehicle Communication Controller (EVCC) and the Battery Management System (BMS) for the V2G session handling and battery management, respectively. We define the trust boundary for the safety functionality of EVCS as a manufacturer-approved combination of these components and perform an integrated threat analysis to determine safety impacts of malicious attacks. Following our analysis, we propose a novel security architecture `secEVCS` that binds trust boundaries for safety and security functionality using trusted computing techniques. `secEVCS` allows the vehicle to access V2G services only if its charging system has the manufacturer-approved configuration. To enforce that, we define the binding between the EVCS components using security policies specified by the OEM that involve both components (e.g., restricting access to EVCC's identity credentials to the BMS's presence), and are verified each time before charging starts. `secEVCS` uses openly specified security anchors, Device Identifier Composition Engine (DICE) for the BMS and TPM for the EVCC. To implement the binding policies, thus, TPM's enhanced authorization feature is employed. This way, `secEVCS` can be protected against the installation of counterfeit spare parts and the reuse of secrets from scrapped components. In order to analyze the associated trade-offs under realistic conditions, the evaluation of our architecture uses a TPM 2.0 chip and ISO 15118 implementation for V2G communication. The outcomes of the performance tests show that the TPM-based binding (using hardware TPM) cannot meet the timing constraints of the chosen protocol without further optimizations with regard to when the binding is verified and how long this verification result holds.

*This contribution is based on a peer-reviewed paper [2] published in collaboration with Dustin Kern, who performed most of the implementation and performance evaluation, Christoph Krauß and Andreas Fuchs, who provided expertise on the TPM 2.0 specification, the open-source software stack, and proposed optimizations. The author made a major contribution to the problem definition, identification of safety-related security threats, and architecture design, as well as management and writing of the paper.*

### Securing Safety-critical Railway Command and Control Systems

As railway CCSs are changed to using Commercial of the Shelf (COTS) products and IP-based communications and the overall connectivity between these systems grows, the risk of malicious attacks targeting the highly-distributed safety-critical railway infrastructure strongly increases, too. However, only the safety certification of hardware and software components of safety-critical applications is mandatory, while security considerations are not part of this process. To acknowledge the need to integrate security controls into safety-critical systems, first domain-specific security standards are developed [66, 67]. Yet, the concepts for co-residing the security functionality on the same hardware platform as the certified safety functionality, which assure freedom of interference between the two areas are lacking. If this property cannot be achieved, the security components must also be certified to high Safety Integrity Levels (SILs), otherwise, the overall certification is voided. Considering the complex and expensive certification procedure and the risk of repeating it after every update, this would hinder the deployment of security measures.

Chapter 7 investigates the safety and security co-engineering for the safety-critical railway CCS. We focus on the security of railway signaling and, in particular, digital Object Controllers (OCs) responsible for the communication with field elements to set safe train routes. We execute a domain-specific requirements engineering process following the German guideline DIN-VDE 0831-104. To provide a more detailed understanding of malicious actors, we perform an extended adversary analysis for this use case. We then use the defined requirements to propose a security architecture for the railway CCS that provides the freedom of interference between applications with different criticality and allows for their joint operation. The architecture consists of a hardware platform with a TPM as a hardware trust anchor, the Multiple Independent Levels of Security (MILS) Separation Kernel (SK), and the required set of security applications co-existing with the safety-critical application, i.e., the OC functionality. The SK ensures that the security applications cannot affect the execution of the safety application in any way. We show that our security architecture meets the security requirements and design a test-bed for the respective simulations. Furthermore, we describe a way to securely integrate Internet of Railway Things (IoRT) applications into the railway CCS based on the proposed architecture.

*This contribution is based on peer-reviewed papers [10], [11], and [9], and a report [24]. These are collaborative works with Markus Heinrich, the team of the research project HASELNUSS of Federal Ministry of Education and Research (BMBF) and the Work Group CYSIS of Deutsche Bahn (DB). The author made major contributions to the security analysis, architecture design, test-bed concept, and definition of the IoRT integration as well as writing of the papers. The prototype of HASELNUSS OC based on this contribution is to be developed and tested in "Digital Test Field Rail" of German DB.*

# 2. Background

In this chapter, we provide some general background relevant for this thesis. Additional background information specific to the individual solutions is included into the corresponding chapters. In Section 2.1, we provide an overview of the established actors and communication protocols in the V2G ecosystem. Section 2.2 summarizes security functionality specified as the open standard TPM 2.0 useful for understanding the proposed contributions.

## 2.1. Communication Protocols in Vehicle-to-Grid Ecosystem

A wide variety of communication protocols exist to provide information exchange necessary for management and coordination of Electric Vehicle (EV) charging sessions and associated energy demands [40, 68, 69]. As a result, a complex infrastructure emerges connecting energy production and delivery facilities, service providers, charging networks, EV fleets, and EV users. The functional scope of this distributed communication infrastructure is largely defined by the following use cases:

**Service Authorization.** To use charging or other services offered at a Charge Point, EV users need to identify and authenticate themselves. This can be as simple as inserting a credit card in the payment terminal (ad-hoc charging), but usually the process involves some kind of identity credential, such as RFID card with the userID, smartphone app with the user account, or an X.509 certificate installed in the vehicle [70]. The authentication protocol can run directly on the charger or require communication with the backend system of the service provider.

**Energy Billing.** When using services provided at a Charge Point, EV users are billed for energy consumed or fed back to the grid by their vehicles. The detailed information about each charging session is collected in the form of Charge Detail Records (CDRs) and transferred to the responsible service provider. B2B billing is also based on this information.

**Smart Charging.** The main goal of smart charging is to maintain grid stability under fluctuating power demands. However, to keep EVs running is equally important for the sustainable transportation. In order to negotiate a grid-friendly and cost-optimal charging schedule, an EV can request information on available services and charging parameters from the Charge Point and inform it about own preferences, e.g., required energy amount, battery capacity, and planned departure time. These data help to distribute energy between plugged-in EVs in such a way that they can achieve their optimal charge state not only without overloading the grid (congestion management [71]) but also while serving as a Distributed Energy Resource (DER) able to react on changing energy supply (grid balancing [72]). During this process, both end-points or the EV user can decide to renegotiate the charging schedule, pause or stop charging.

**Charging Session Monitoring and Control.** The Charge Point measures the consumed energy and periodically provides meter readings both to the charging EV and to the service or energy provider to enable monitoring of the charging status. The EV can pause charging and resume it under the previously negotiated conditions. Similarly, the service provider may decide to stop the session remotely or initiate the renegotiation.

**Credentials Management.** Identity credentials stored on the vehicle or on the charger, can be installed, updated, or invalidated (revoked) in an automated way using the same communication path as the charging service.

The corresponding communications system architecture usually consists of the following main components [73, 74, 40, 75]:

**Electric Vehicle.** The EV is a battery-powered vehicle able to communicate with the Charge Point using High Level Communication (HLC). Once the EV is connected to the Charge Point, it will try to communicate with the Charge Point to start the charging process.

**Charge Point.** The Charge Point supplies the EV with electricity delivered from an energy provider and Distribution System Operator (DSO) necessary to recharge its battery. The Charge Point contains one or several meters to measure EV's power consumption during a charging session. Each Charge Point supports at least one socket that allows to connect an EV and provide power to charge it. The Charge Point may also support several plugs with different formats that may also allow to charge more than one EV at a time in parallel. The Charge Point is connected to a Charge Point Operator (CPO) that is responsible for its installation, management, and maintenance.

**Original Equipment Manufacturer.** The Original Equipment Manufacturer (OEM) is the manufacturer of the EV and installs the required components during the production of the vehicle. It can also install the necessary identity credentials before the vehicle is delivered to the user.

**Charge Point Operator.** The CPO is the entity that operates and manages charge points. The CPO supports its Charge Points during identification of EVs and authorization of charging sessions if necessary. To do that, the CPO may forward the EV's authentication data to Contract Clearing House (CCH) and/or the EV's E-Mobility Service Provider (eMSP). This implies some kind of a contract agreement between these parties. If the response is positive, the CPO may also start charging remotely or forward the permission to the Charge Point. Being the only entity that has access to the Charge Point, the CPO also collects CDRs regarding conducted charging sessions or information necessary to produce such CDRs (meter values, IDs, timestamps, etc.) that are used in B2B (among service providers) and B2C (among eMSP and its customers) billing. In order to supply its Charge Point with electricity, the CPO depends on services of DSO that connects Charge Points to the power grid. Based on information about the available amount of energy, the CPO sets constraints for each Charge Point and can limit the energy consumption of EVs during charging sessions.

**EV User / Customer.** The customer is the EV user and charges an EV based on a charging service contract with an eMSP. The customer can be a person or a company that owns a fleet of vehicles used by its employees. In both cases the owners are also the owner of these EVs.

**E-Mobility Service Provider.** To validate if the EV user is allowed to charge the user identifier provided by the Charge Point is verified with the eMSP. The eMSP is a charging service provider that has a contract relationship with the EV user that allows to charge the EV at a Charge Point. The Charge Point must either belong to the eMSP or it must support roaming (via CCH). The eMSP communicates with CPOs of the Charge Points used by its customers in order to collect CDRs containing the amount of the energy consumed by EVs during charging sessions and bill their users for the charging service.

**Contract Clearing House.** The CCH serves as an intermediary between CPOs that provide Charge Points and eMSPs that provide EVs with access credentials necessary to use charging services. In case a CPO cannot verify a particular customer it can use a roaming service of the CCH to identify an eMSP that has a contract with this customer and will compensate the charging costs. In order to provide this service, the CCH may collect and store lists of customer identifiers of the partner eMSPs.

**Distribution System Operator.** The DSO is responsible for the planning, operation, maintenance, and the development of the distribution network, the quality of the electricity supply (power delivery, voltage stability etc.), and the delivery of electricity to consumers, such as Charge Points [76]. Since it is also responsible for the voltage stability in the distribution network, the DSO may also limit charging rates or overall energy available at a certain node.

### 2.1.1. ISO 15118 Protocol: Connecting Electric Vehicle and Charge Point

ISO/IEC 15118 is an international standard that specifies a communication interface between an EV and the power grid for bidirectional power transfer. The standard is developed by Joint Working Group (JWG) 1 of IEC Technical Committee (TC) 69 together with Sub Committee (SC) 31 of TC 22 on road vehicles of the International Organization for Standardization founded in 2009.

The specification consists of several parts defining the communication protocol on the different levels of the Open Systems Interconnection (OSI) Model. In the context of this work, only the parts ISO 15118-1: “General information and use-case definition” [37] and ISO 15118-2: “Network and application protocol requirements” [38] are relevant. In this specification, the HLC is described, i.e., communication using the standard network stack, in contrast to the signaled charging based on IEC 61861-1 [38]. ISO 15118 specifies a request-response protocol including corresponding message and sequence requirements, data models, and XML/EXI-based data representation formats.

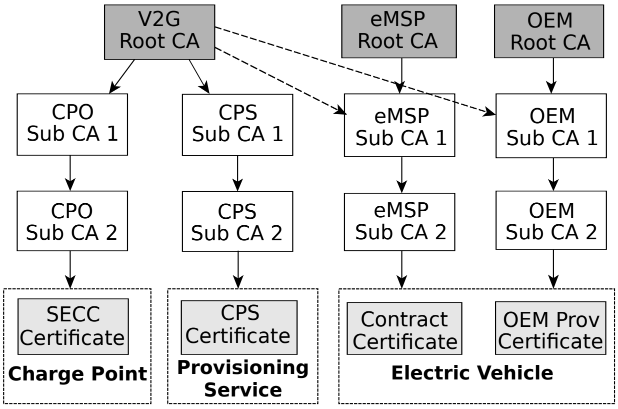


Figure 2.1: PKI Hierarchy for PnC (adapted from the ISO 15118 [38])

The standard focuses on the communication between the Electric Vehicle Communication Controller (EVCC) and the Supply Equipment Communication Controller (SECC) in the EV and the Charge Point, respectively. The standard also introduces the role of Secondary Actor (SA) to represent various backend systems including OEM, CPO, eMSP, and Certificate Provisioning Service (CPS). It addresses major Vehicle-to-Grid (V2G) use cases, such as basic EV charging, smart charging, certificate handling and Value Added Services (VAS) [37]. ISO 15118 supports two modes for authenticated charging sessions: Plug-and-Charge (PnC) and External Identification Meanss (EIMs). The concept of PnC fully automates EV identification and authentication, charging authorization and billing of EV users for the energy consumed during charging sessions. Instead of actively authenticating themselves to a charging service provider using RFID cards or smartphone apps referred by the standard as EIM, users only need to plug the EV into a CP's socket. To activate charging and start receiving energy, the EV uses own identity credential issued by an eMSP and installed in the vehicle, and does not require any further actions from the user. Therefore, the standard is being actively promoted by OEMs and service providers to simplify charging for their customers [77].

The concept of VAS allows for any arbitrary functionality outside the scope of ISO 15118. The standard considers such applications as reservation of Charge Points or Internet access for infotainment during waiting time [37]. In the latest edition ISO 15118-20 [39], VAS are used together with wireless charging and robotic connection devices to enable automated parking and positioning.

In terms of security, ISO 15118 deploys Transport Layer Security (TLS) to protect the PnC and VAS communication in public environments. Additionally, digital signatures (using XML Signature) are used to authenticate messages. The respective credentials are provided by the V2G Public Key Infrastructure (PKI) (see Annex E in [38]).

**ISO 15118 Trust Model.** The ISO 15118 standard uses the dedicated X.509v3 PKI shown in Figure 2.1 to support trust establishment. The root of trust in this PKI is the so-called V2G root Certificate Authority (CA) responsible for authenticating entities in the EV charging and billing ecosystem. The standard assumes that each region of the world has one trust center for e-mobility that acts as the V2G root CA in this region. The certification of subordinate CAs of CPO and CPS by the V2G root CA is mandatory. OEM and eMSP can operate their own root and sub-CAs that issue and sign credentials for EVs or use the V2G root CA instead. End entity certificates can only be issued by a sub-CA. The certificate path length is limited to three certificates. In Figure 2.1, we use dark grey color to depict the root certificates. The dashed lines show the optional certification of OEM and eMSP root CAs by the V2G root. The certificates at the bottom colored in light grey are carried by end entities.

*SECC Certificate* containing a unique Charge Point Identifier (CPID) is issued to a Charge Point by the CPO and is used during TLS communication setup with an EV for server authentication. Thus, the respective V2G root certificate needs to be stored in an EV to enable PnC. These certificates have a short validity period of two to three months.

An EV possesses two digital certificates. The *OEM Provisioning Certificate* is a long-term identity assigned to an EV by its OEM during manufacturing. This certificate has a very long validity period (around 40 years) and is usually never changed. This certificate contains an identifier Provisioning Certificate Identifier (PCID) required for the registration with an eMSP and provisioning of the contract certificates. The *Contract Certificate* is bound to a charging contract with an eMSP and allows an EV to use PnC at a Charge Point. The respective contract ID or E-Mobility Account Identifier (EMAID) is part of the contract certificate and provides a basis for billing the EV user for charging sessions. The

EMAID comprises of a prefix, which encodes a short identifier of the eMSP, and a contract number. Contract Certificates have a short validity period (between four weeks and two years).

The *CPS Certificate* belongs to the Certificate Provisioning Service (CPS) that delivers contract certificates to EVs during their enrollment for the charging service. This provisioning service can be hosted by an eMSP or any other actor. Its only purpose is to establish trust in data received by an EV from an eMSP in case eMSP root CA is not known to the EV. The eMSP forwards the newly created contract data package for validation and provisioning to the CPS. The CPS validates contract data packages and signs them. If the EVCC is connected to the SECC and has no valid contract credentials, the EVCC can automatically request the contract data packages from the CPS via the ISO 15118 connection to the SECC.

In the following, we provide an overview of the ISO 15118 protocol flow according to the use cases defined in [37].

**Communication Setup.** The EVCC initiates the communication with the SECC. First, a TLS session is established. Then, EVCC and SECC negotiate the protocol version using the *supportedAppProtocolReq*. Next, the EVCC sends the *SessionSetupReq* containing its unique *Electric Vehicle Communication Controller Identifier (EVCCID)* to the SECC to negotiate a session ID. Once the session is established, the EVCC requests the list of services offered by the Charge Point using the *ServiceDiscoveryReq*, selects required services and sends a *PaymentServiceSelectionReq* indicating the payment option used for the services. In our case, the payment option chosen is the PnC option called *Contract*, which indicates the use of Contract Certificates.

**Identification, Authentication, and Authorization.** After the EV selected the payment method, it negotiates the payment details using the *PaymentDetailsReq*. This request contains *EMAID*, *Contract Certificate*, and the respective certificate chain of the EV. The SECC responds with a challenge. The EVCC signs the challenge using the private key of the Contract Certificate and sends an *AuthorizationReq* with the signed challenge to the SECC requesting authorization to start a charging session. Once the EVCC receives a positive *AuthorizationRes*, this sequence is finished.

**Target Setting and Charge Scheduling.** First, the EVCC may request the SECC to get specific tariff tables for the given *EMAID* and charging mode from the eMSP using the *ChargeParameterDiscoveryReq*. After the EVCC receives the schedules, it sends a *PowerDeliveryReq*. If the SECC confirms, the Charge Point will provide voltage to the power outlet and the EV starts charging.

**Charge Control and Rescheduling.** During this sequence the EVCC periodically sends the *ChargingStatusReq*. The response message of the SECC may contain the *ReceiptRequired* flag triggering the EVCC to request a metering receipt. This message also contains the current *meter information* and the *Electric Vehicle Supply Equipment Identifier (EVSEID)*. To request a receipt the EVCC sends a *MeteringReceiptReq* containing the *SessionID*, current charging schedule, and the signed meter values it previously received.

**End of charging process.** The EVCC may first send a *PowerDeliveryReq* message to stop the SECC providing voltage to the power outlet using the appropriate *stop* indicator for the *ChargeProgress* parameter.

**Changes in the Upcoming Edition.** In Q2 2022, a new edition of the part 2 ISO/FDIS 15118-20 [39] will be published. This version contains significant changes to message formats and sequences and is no longer fully backward compatible. New ISO 15118-20 extends the scope of the series to include future-oriented use cases, such as automated connection device, bidirectional power transfer, wireless communication for conductive charging and wireless charging. Moreover, some processes also change, including the provisioning of the contract certificates being only possible using the OEM provisioning certificate and multiple contract certificates for various charging contracts being supported.

ISO 15118-20 also improves on the security concept to address previously reported problems [50]. Starting with this edition, mutually authenticated TLS is mandatory for all use cases (for this, a Vehicle Certificate has been introduced), the cipher suites and key length are updated to meet the requirement till 2030+, and an option for cryptographic agility is provided. Table 2.1 provides an comparison of cryptographic algorithms used in ISO 15118-2 and ISO 15118-20.

Table 2.1: Cryptographic Algorithms in ISO 15118-2 and ISO 15118-20

| Function | ISO 15118-2 | ISO 15118-20 |
|---|---|---|
| TLS version | TLS 1.2 and higher | TLS 1.3 |
| TLS cipher suites | TLS_ECDH_ECDSA_AES_128_CBC_SHA256<br>TLS_ECDHE_ECDSA_AES_128_CBC_SHA256 | TLS_AES_256_GCM_SHA384<br>TLS_CHACHA20_POLY1305_SHA256 |
| Signature algorithm | ECDSA (secp256r1, SHA256) | ECDSA (secp521r1, SHA512) & EdDSA (Ed448, SHAKE256) |
| Hash function | SHA256 | SHA512 & SHAKE256 |
| Key exchange | ECDH/ECDHE with secp256r1 | ECDH/ECDHE with secp521r1 & X448 |
| Symmetric cipher | AES-CBC-128 | AES-GCM-256 & AES-CFB-256 |
| Elliptic Curve | secp256r1 | secp521r1 & Curve448 |

**Certificate Handling with the Application Guideline VDE-AR 2802.** The application guideline VDE-AR 2802 [78] of the German standardization organization DKE[1] addresses specification gaps of ISO 15118-2 [38] regarding the exchange of digital certificates across backend actors and defines technical measures for installing contract credentials using alternative communication paths such as the OEM proprietary telematics link. The guideline enables the intermediate storage of the information relevant for the creation and provisioning of contract credentials in OEM Provisioning Certificate Pool and Contract Certificate Pool, which can be accessed by any actor in the V2G ecosystem. The OEM Provisioning Certificate Pool enables the data exchange between OEMs and eMSPs. After an EV is prepared for delivery to its user, the OEM uploads vehicle's identification data including OEM provisioning certificate into this Pool, where they are stored under the respective PCID. When the eMSP receives a PCID from the user, it can directly extract all information necessary for generating a contract credential for the user's EV from this OEM Provisioning Certificate Pool. Having created the contract credential for the user's EMAID, the eMSP transfers it as part of the contract credential data package to the CPS for the verification and authentication (signing). After being signed, the package is uploaded to the Contract Certificate Pool, where it is stored under the EMAID, the PCID, and the protocol version.

The Contract Certificate Pool enables the data exchange between the CPO/SECC and the CPS as well as the OEM and the CPS. When the SECC receives a certificate installation request from the EVCC, it can download the respective contract credential from this Pool using the provided PCID and deliver it to the vehicle without delay. Alternatively, the OEM can collect contract credentials for own vehicles from the Contract Certificate Pool and deliver them via the telematics system to the EVs. Similarly important is the performance gain for the eMSP, since contract credentials do not need to be created on-the-fly when a request with the necessary information about the vehicle arrives but can be pre-generated using the information from the OEM Provisioning Certificate Pool. This helps meet strict timing constraints of ISO 15118 for the delivery of contract credentials.

### 2.1.2. Backend Communication Protocols

Though a Charge Point often physically is an end point of the communication between an EV and the V2G infrastructure, logically it is seldom the case. In order to connect the Charge Point to the various actors in the backend and enable the information exchange between the vehicle and actual service providers like eMSP or DSO, ISO 15118 data or messages are forwarded via purpose-built backend communication protocols. In this section, we provide an overview of the open communication protocols necessary to implement the ISO 15118 use cases for charging authorization and billing.

**OCPP Protocol: Connecting Charge Point and its Operator.** One of the most important protocols for the communication between a charger and the backend system of the responsible CPO is Open Charge Point Protocol (OCPP) developed by Open Charge Alliance (OCA), an international organization with over 220 members[2]. This open and freely available

[1] DKE German Commission for Electrical, Electronic & Information Technologies of DIN and VDE, `https://www.dke.de/en`
[2] `https://www.openchargealliance.org/`

communication protocol is adopted in more than 50 countries including in Asia, Europe and parts of the USA[3] and is considered de-facto industry standard [40]. Furthermore, there is an ongoing effort to convert OCPP into an official international standard IEC 63110 to provide a formally interoperable set of features for operation and management of EV charging and discharging infrastructures [76, 79].

The standard has over 10 years of the development history, with OCPP 1.6 [80, 81, 82] released in 2015 being presently prevalent in the EV charging ecosystem. The latest versions are OCPP 2.0 [83, 84] from 2018 and its successor OCPP 2.0.1 [85, 86] from 2020, which already see first adoptions [87, 88]. Compared to OCPP 1.6, the 2.0.x versions introduce a lot of improvements with regard to communication security, smart charging, and direct support of ISO 15118.

OCPP is an application layer request-response protocol, where operations can be initiated by both the Charge Point and the operator's backend (also called Central System (CS) or Charging Station Management System (CSMS)). OCPP operations support various communication infrastructures, with OCPP 1.6 providing specifications for implementations using Javascript Object Notation (JSON) over WebSockets [81] or Simple Object Access Protocol (SOAP) over HTTP(s) [81]. Starting with OCPP 2.0 only JSON/WebSockets format is supported [84, 86]. This communication occurs over Wide Area Network (WAN) (often cellular) and uses the standard TCP/IP stack. In the following, we review the provided functionalities and discuss the security status quo for these most relevant OCPP versions.

OCPP 1.6 operations are embedded into the following 6 functional groups or profiles, each of which can be implemented independently [80]:

**Core:** This profile contains basic management functions, such as Charge Point configuration, diagnostics (boot status, heartbeat, and status information), and remote command execution (start/stop transaction, reset, and unlock connector). The basic charging operations are also included in this profile: a charger can start or stop a charging session (transaction), after requesting the authorization status for the EV from the CSMS using a unique identifier (idTag) provided by the vehicle itself or its user. In response, the backend system confirms if the provided identifier is accepted or sends a reason for rejection. The Charge Point may store the successfully verified identifier in the local authorization cache to reduce interactions with the backend. To inform the CPO backend about the charging status and to enable billing of the consumed energy, the Charge Point can periodically send meter values to the backend, together with the transaction ID or between charging sessions. Furthermore, OCPP allows vendors to extend the OCPP message set and thus to exchange arbitrary data between the charger and the backend.

**Firmware Management:** This profile enables the management of firmware updates and diagnostic logs. In order to update a Charge Point, the backend sends a link to the update package, which the charger retrieves and installs by itself. The Charge Point then can notify the backend about the update status. Logs are collected in a similar way: the Charge Point receives an upload link to provide the logs and notifies the backend, when this operation is finished.

**Local Authorization List:** This profile allows to manage authorization lists stored locally on a Charge Point to reduce network traffic with the CPO backend or if the connection is not reliable. The Charge Point can simply compare the provided user identifier with the local whitelist and grant the authorization without asking the backend. This list is provided and refreshed by the backend system.

**Reservation:** This profile allows reserving a connector on the Charge Point starting from the time of request (not for the future) or cancel the previously made reservation.

**Smart Charging:** Basic smart charging functionality uses so-called charging profiles to set the maximum power or current available for a Charge Point, a selected connector, or a selected transaction during a time interval, i.e., create a charging schedule. Charging profiles can be combined to describe the desired consumption behavior over time and/or depending on the energy availability and price. Smart charging can be used to implement various managed charging strategies, such as load balancing (energy distribution) between the Charge Point's connectors[4], centrally managed charging schedules to match grid capacity and reduce peak demands, or dynamic adjustment of power consumption by a local controller (e.g., for several Charge Points at a parking lot or in smart home applications). The calculation of a charging schedule is highly complex with optimization goals varying for residential and public charging, for renewable energy resources, or for individual requirements and preferences [36, 89, 90, 91].

[3] `https://www.openchargealliance.org/about-us/appraisal-ocpp/`

[4] For example, load balancing is performed by EVBox `https://evbox.com/uk-en/learn/faq/difference-priority-load-balancing`

**Remote Trigger:** CPO can remotely trigger a charger to send certain messages, which otherwise are initiated by the charger. These include various status messages and meter data.

OCPP 1.6 lacks a profile containing security-related operations. The implementation specifications OCPP 1.6-J [82] (using JSON/Websockets) and OCPP 1.6-S [81] (using SOAP) recommend to use TLS 1.2 [92] to encrypt the connection between a Charge Point and the backend in the absence of a trusted network. Thereby, verifying the backend's identity is considered optional (cf. Section 6.2.1 of [82]). To authenticate a Charge Point, the HTTP basic authentication scheme [93] (over TLS) is recommended, without specifying any protection means for the respective credentials stored on the charger. Consequently, the OCPP 1.6 security analyses [94, 95] concluded that Man-in-the-Middle (MitM), impersonation, and data tampering attacks have a very high potential to cause energy theft, disrupt energy services, or even overload grid. Moreover, since most of the transferred data is Personally Identifiable Information (PII), such as customer identification and charging location, (personalized) energy tariffs, meter readings reflecting a unique charging curve [96], these attacks also affect privacy and lead to violations of the General Data Protection Regulation (GDPR).

As we show below, OCPP 2.0.x improves the security concept. The introduced enhancements has been adapted to OCPP 1.6-J in [97], to close the security gaps in this version, too. The protection measures include secure connection setup using TLS 1.2 with unilateral (server-side) or mutual authentication; logging and reporting of security events; and secure Over-the-Air (OTA) firmware updates. On the downside, it is allowed for legacy systems to downgrade to now deprecated TLS 1.0/1.1 (and corresponding cipher suits). In order to save the effort of checking the revocation status for server certificates, fast expiration for the backend system certificates with maximum 24 hours of validity is used. This is not valid for Charge Point certificates, which can be updated via OCPP using standard Certificate Signing Requests (CSRs). Minimum protection for the respective private key is also mentioned, i.e., it is required that the key never leaves the Charge Point and is not accessible via a communication interface. Some security functions lack the definition, under which security profile they can be executed, i.e., it may be assumed that security logs and firmware updates can be transferred in clear text. The proposed measures partly build on the end-to-end security design from [98, 99], which aims to ensure integrity of meter readings, enable secure communication channels, and provide strong customer authentication. However, the evidence necessary to support the non-repudiation claims (cf. ISO/IEC 13888 [100, 101]) was not considered, and neither the privacy concept described in [98] to protect sensitive data outside the secure channel against intermediaries.

OCPP 2.0.x [83, 85] is a complete overhaul of OCPP 1.6 with improved and new functionalities and is not backward compatible. The specification employs more fine-grained 16 functional blocks (A to P) [85, 102] instead of the 6 profiles used in OCPP 1.6 [80] to accommodate all management use cases.

The most interesting for this thesis are the added security features (TLS and Charge Point certificate management) and support for PnC and smart charging (V2G) specified in ISO 15118 [38][5]. The ISO 15118 and security functionalities are distributed across several functional blocks as the following list shows.

**A. Security (New):** Beside the usage of a trusted network (VPN), the new OCPP security concept allows to use a TLS channel to securely transfer application layer messages between a charger and the CPO backend. These options are represented as 3 OCPP security profiles (cf. Section 1.3 in [102]). The first profile uses HTTP basic authentication for the Charge Point and relies on network security, while the second and the third use TLS with server-side authentication (and HTTP basic authentication for the client inside the secure channel) or with mutual authentication, respectively. This functional block supports the security profiles by enabling the update of Charge Point's authentication credentials, i.e., passwords for basic authentication or TLS certificates. Since the SECC certificate from ISO 15118 can also be used as the Charge Point certificate, the corresponding functions also need to be used to update SECC certificates. Further security functionality realized in this block is logging and reporting of security events from a pre-defined criticality-based list.

**B. Provisioning (Extended):** This block together with Block N supports on-boarding, configuration, and monitoring of Charge Points. The management functions are based on the newly introduced 3-tier[6] Device Model (cf. Section 4 in [103]) that allows the CPO to address various components and variables of any charger model using this generalized representation.

**C. Authorization (Extended):** Beside RFID- and whitelist-based options previously available to authenticate users and authorize EV charging sessions, the support of further existing authentication and payment methods is added. The

[5] The upcoming edition ISO 15118-20 [39] is not supported but the corresponding effort to extend the OCPP specifications has been started.

[6] The tiers are Charge Point, Electric Vehicle Supply Equipment (EVSE), and Connector

new options include payment via the service provider backend using smartphone apps, using credit/debit cards at payment terminals (ad-hoc charging), or physically unlocking the charger with a button or a key. Moreover, the authentication methods required in ISO 15118, i.e., PnC and EIM are also integrated in this functional block. In the case of PnC, the EMAID is extracted from the ISO 15118-2 request message and forwarded to the CPO backend via OCPP.

**D. Local Authorization List Management:** In this block, functions are identical to the respective profile in OCPP 1.6.

**E. Transactions (Extended):** The new functions provide a fine-grained status of charging transactions and allow to integrate pausing and stopping of ISO 15118 charging sessions.

**F. Remote Control (Extended):** The previously available functions for remote control and remote triggering of events are extended with remote support for ISO 15118-2.

**G. Availability:** As in the previous version, the status notifications are used to ensure the availability of a Charge Point.

**H. Reservation:** Charge Point reservation functions also remain unchanged. In [104], an OCPP extension is proposed that enables the negotiation of charger reservations starting at a future time and accounting for user's preferences and flexibility.

**I. Tariff and Cost (New):** This new functional block allows Charge Points equipped with a display to show EV users tariff and cost information related to their current transaction. Consumer protection regulations often require to display the total recharge cost before the charging starts and a summary of the charging process (time, consumed energy, tariff and cost) after it stops.

**J. Metering (Extended):** ISO 15118-2 [38] provides an option for an EV to request the periodic reporting of meter values by the Charge Point to monitor the charging progress, whereby the vehicle needs to return a signed confirmation that the readings were received. OCPP integrates the metering information exchange in this block.

**K. Smart Charging (Extended):** The original OCPP design only considered Charge Points directly connected to the CPO backend (CSMS). To reflect the recent developments in energy management, this design is extended in the new version to support other topologies and use cases involving an Energy Management System (EMS), local controller, or proxy. Moreover, ISO 15118-2 allows EVs and their users to actively participate in smart charging and (re)negotiate charging profiles based on current EV preferences (recharge energy and departure time). For this purposes, a new mechanism to negotiate and manage charging profiles is added that allows, among other things, changing the profile during an active transaction.

**L. Firmware Management (Extended):** This block enables secure OTA firmware updates for Charge Points. The update packages are signed by the CPO in such a way that a charger is able to verify their authenticity and integrity. In addition, the updates can be distributed via a local controller managing a set of Charge Points, which are not connected directly to the CPO backend.

**M. ISO 15118 Certificate Management (New):** This new functional block enables the installation and update of contract credentials for an EV/EVCC according to ISO 15118-2 [38]. The respective requests are forwarded by the Charge Point to its CPO using OCPP. Also Online Certificate Status Protocol (OCSP) responses necessary for the validation of the Charge Point certificate chain during the PnC connection establishment can be retrieved using the functions in this block.

**N. Diagnostics (Extended):** Beside uploading a diagnostics file from a Charge Point, new functions in this block allow the CPO to manage the monitoring process using the Device Model (cf. Section 4 in [103]), i.e., define parameters and thresholds to monitor and set alarms.

**O. Display Message (New):** This block enables the CPO to display arbitrary messages to users, i.e., actual tariffs, personalized messages to registered clients, etc.

**P. Data Transfer (Extended):** Unlike ISO 15118, OCPP is easily extensible and allows adding custom extensions to support new functionalities. In this version, even more flexibility is added to exchange arbitrary data or messages using `DataTransferRequest` message or to add custom elements/attributes to a standardized JSON schema using `CustomData` element.

At the moment of writing, OCPP 2.0.x is the only backend protocol that natively supports ISO 15118-2 [38] and enables processing of EV requests in the backend. However, since OCPP only connects Charge Points to their operators, to perform the authorization and billing, the data need to be forwarded to the eMSP concerned with the particular vehicle. With IEC 63119 [105] being work in progress, there is currently no official standard for this communication path. In the following, we provide an overview of the existing communication protocols developed by the industry to realize the aforementioned services.

**Roaming Protocols: Connecting Backend Systems.** In the present condition of the charging infrastructure, an EV cannot communicate directly with the eMSP, which enables the vehicle's access to V2G services based on the contract with its user. Therefore, to provision ISO 15118 contract credentials, authorize a charging session and bill the user for the consumed energy, the CPO needs to forward the corresponding information to the responsible eMSP. E-mobility roaming services realized with custom roaming protocols connect CPOs and eMSPs so that EV users can charge their vehicles at Charge Points owned by any CPO that has a direct agreement with their eMSP or shares a roaming platform provided by the same CCH.

In contrast to OCPP, there are several roaming protocols being simultaneously developed and deployed by various CCHs to offer roaming services to customers registered on their platforms [40, 106]. Below we list those roaming protocols that have freely available specifications and are maintained by major market players.

**Open Intercharge Protocol:** Open Intercharge Protocol (OICP) 2.3 [107, 108] (JSON/REST) is developed by Hubject[7] to connect CPOs and eMSPs to their roaming platform. The latest version of the protocol is released as open source.

**Open Clearing House Protocol:** Open Clearing House Protocol (OCHP) 1.4 [109] (SOAP) is developed by Smartlab[8] and ElaadNL[9] for their CCH e-clearing.net.

**OCHPdirect:** OCHPdirect 0.2 [110] (SOAP) is an extension of OCHP described above, which allows CPO and eMSP to establish a direct connection.

**Open Charge Point Interface:** Open Charge Point Interface (OCPI) 2.2 [111] (JSON/REST) is currently maintained by the Netherlands Knowledge Platform for Public Charging Infrastructure (NKL), incorporating profit and non-profit organizations. Thus, OCPI is an independent protocol. OCPI provides both direct connections between CPO and eMSP and the usage of a roaming platform.

**eMobility Protocol Inter-Operation:** eMobility Protocol Inter-Operation (eMIP) 0.7.4 [112] (SOAP) is created by GIREVE[10], a roaming service provider in France.

Though the above roaming protocols are not interoperable, they provide similar functionality, especially, with regard to authorization and billing support. Thereby, only OCHP and OICP support user authorization using ISO 15118's EMAID. For example, the authorization service using OICP offers three options: (i) offline authorization (by the CCH) based on the contract information stored on the Hubject platform; (ii) using the eMSP identification extracted from the authorization request provided by the CPO; or (iii) platform-wide broadcast of the request to find the responsible eMSP. Therefore, this service must be implemented by both CPOs and eMSPs using the respective part of the specification. In the case of ISO 15118's PnC, the authorization request contains the EMAID, which prefix is used by the CCH to determine the correct eMSP and forward the request. For all charging sessions started via the roaming service, the CPO must send a Charge Detail Record (CDR) with detailed information including EVSEID, EMAID, timestamps, meter values and consumed energy to the CCH when charging stops. This information is forwarded to the eMSP for billing and stored by the CCH. OICP enables further typical roaming services such as the reservation of Charge Points, sharing of detailed Charge Point information including current tariffs via the CCH platform, and monitoring the status of charging sessions for more transparency between the CPO and eMSP.

[7] https://www.hubject.com/

[8] https://smartlab-gmbh.com/

[9] https://www.elaad.nl/

[10] https://www.gireve.com/

## 2.2. Selected Functions of Trusted Platform Module 2.0

The Trusted Platform Module (TPM) 2.0 is a secure co-processor that acts as the core security enforcement point in the computer system [113]. The TPM 2.0 specification [114] developed by the Trusted Computing Group (TCG) defines the functionalities for building TPMs for different platforms. In 2015, it was adopted as the international standard ISO/IEC 11889 [115]. Beside standard Hardware Security Module (HSM) functionality of secure key storage and secure execution environment for cryptographic operations, the TPM 2.0 provides additional security mechanisms allowing to limit the access to the cryptographic keys, such as direct import of encrypted keys, measured boot, enhanced authorization, and credential protection.

During measured boot, hash values are calculated for firmware components included into the boot chain and are stored in Platform Configuration Registers (PCRs) to verify the integrity of the firmware. These measurements can be used in policies, e.g., to seal keys to a particular firmware state [113]. Enhanced authorization allows associating arbitrary policies with all TPM objects that must be satisfied in order to authorize an action on that object (cf. [114], Section 19.7). The policies can be simple password authentication, time-based conditions, or internal counter values [113]. Credential protection allows credential providers to ensure that the credential issued for a key can only be used by a system with the authentic TPM (cf. [114], Section 24).

The TPM 2.0 specification provides cryptographic algorithm agility by supporting a list of algorithm identifiers, which can be chosen from the TCG's Algorithm Registry [116]. The TPM supports asymmetric algorithms such as RSA and ECC but also symmetric algorithms such as AES, HMAC, or hash functions such as SHA1 or SHA2.

The TPM is mostly realized as a dedicated hardware chip and provides a specially shielded area for secure storage and usage of keys, which makes it very hard to extract, copy or duplicate the keys from the TPM. However, since the TPM's internal NV memory is very limited, keys are usually stored as encrypted TPM objects on the host system and are only decrypted within the TPM before usage. Keys can also be generated outside of the TPM and then be imported into the TPM protected object hierarchy using key duplication mechanisms (cf. [114], Section 23). Most TPM chips have a security certification, e.g., Common Criteria (CC) EAL4+, and some are also qualified according to the automotive AEC-Q100 standard. But implementations as a a System on a Chip (SoC) or as a firmware TPM are also possible, with reduced security guarantees. From the implementation perspective, the TPM's advantage is the availability of the hardware and the availability of the open source TPM Software Stack (TSS)[11].

**TPM 2.0 Commands.** Table 2.2 provides an overview of the TPM 2.0 commands relevant for the solutions presented in this work. The complete list of commands and corresponding Application Programming Interface (API) descriptions is specified in [117].

Table 2.2: Relevant TPM 2.0 Commands

| Command | General usage | Usage in thesis |
|---|---|---|
| `TPM2_CreatePrimary` | The command creates a TPM primary key using the provided key template (cf. [117], Section 24.1). | The command is used during creation of the TrustEV key hierarchy to generate the Storage Root Key (SRK). As input the key template is provided, i.e., key type (ECC key), key length, and attributes. |

[11] https://github.com/tpm2-software

| Command | General usage | Usage in thesis |
|---|---|---|
| TPM2_Create | The command creates a TPM object, e.g., a cryptographic key using the provided key template (cf. [117], Section 12.1). | The command is used to create OEM provisioning and contract credential keys in the TPM key hierarchy using the key template, i.e., key type (ECC), key length, and attributes, and usage authorization policy. If an Original Equipment Manufacturer (OEM) defines an enhanced authorization policy to restrict the usage of the created key, it also needs to be provided during the creation of this key. The output contains the public area (including public key) and the encrypted private area (private key) that can be loaded into the TPM to perform cryptographic operations. |
| TPM2_EvictControl | The command is used to store a key in the on the TPM in persistent memory (cf. [117], Section 28.5). | The command is applied to the SRK of the TrustEV key hierarchy. |
| TPM2_Load | The command is used to load a key (both public and private part) into the TPM using the loaded parent key to decrypt the private part (cf. [117], Section 12.2). If only a public key needs to be loaded, the command TPM2_LoadExternal is used instead. | The command is used to load OEM provisioning and contract credential keys. |
| TPM2_ActivateCredential | The command is used within the credential protection protocol (cf. [114], Section 24) and allows to bind a credential (usually an X.509 certificate) to a TPM key (cf. [117], Section 12.5). When a credential provider issues a certificate for a key allegedly created in a TPM, this command can be used by the credential provider to encrypt the certificate using the TPM's known primary key in such a way that it can be recovered only if the TPM can really load the key (with the denoted attributes), for which the certificate has been requested. The detailed protocol can also be found in [113], Chapter 9, "Activating a Credential". | The command is used in HIP and HIP 2.0 during the enrollment of the contract keys generated in the vehicle's TPM to provide the security assurance to the E-Mobility Service Provider (eMSP). |
| TPM2_Import | The command is used to import keys into the protected key hierarchy of the TPM (cf. [117], Section 13.3). Thereby, a key is symmetrically encrypted based on the TPM's storage key and can be loaded into the hierarchy and used afterwards. This command is part of the duplication functionality, where an object can be stored underneath additional storage keys of the same or a different TPM (cf. [114], Section 23.3). In the work, the encryption method "Outer Duplication Wrapper" described in [114], Section 23.3.2.3 is used. | The command is used in TrustEV to import contract keys generated by the eMSP into the vehicle's TPM. |
| TPM2_VerifySignature | The command is used to validate a signature using a loaded key and a provided digest (cf. [117], Section 20.1). | The command is used e.g. to verify signature on the contract credential data package and during policy validation. |
| TPM2_Sign | This command calculates a signature using an a provided hash and a signing key (cf. [117], Section 20.2). | The command is used to sign, e.g., certificate installation request using private OEM provisioning key or an authorization challenge using private contract key. |

| Command | General usage | Usage in thesis |
|---|---|---|
| `TPM2_HMAC` | This command is used to calculate an Hash Message Authentication Code (HMAC) over provided data (cf. [117], Section 15.5). | This command is used e.g. to calculate HMACs in Plug-and-Patch (PnP) protocol. |
| `TPM2_NV_DefineSpace` | This command is used to define an Non-Volatile (NV) index and to allocate space for storing the associated data (cf. [117], Section 31.3). The NV index can belong either to the storage hierarchy (user-owned) or of the platform hierarchy (OEM-owned) of the TPM. In this work, only OEM-owned indices are used. | The command is used together with the PolicyNV. |
| `TPM2_NV_Increment` | This command is used to increment the counter value of an NV index with the respective attribute (cf. [117], Section 31.8). | The command is used, e.g., in `secEVCS` to invalidate an old Battery Management System (BMS) together with the PolicyNV. |
| `TPM2_StartAuthSession` | The command is used to start a policy (authorization) session necessary to satisfy the policy (cf. [117], Section 11.1). The policy assertions are subsequently satisfied (in the order they were created) by invoking the corresponding TPM commands. | This command is used together with every policy command. |
| `TPM2_PolicyAuthorize` | This command is part of the TPM's Enhanced Authorization (EA) functionality and is used to change the policy associated with a TPM object, e.g., a key after this object was created (cf. [117], Section 23.16). This command is included after the policy that may need to be modified later on and contains the reference to a public key that can sign an update for this policy (cf. [114], Section 19.7.11). This, e.g., helps avoid key locks if the PCR's value the key is sealed to changes after a firmware update (see also [113]). | This command can be used by the OEM to update policies for Vehicle-to-Grid (V2G)/Plug-and-Charge (PnC) keys. |
| `TPM2_PolicyNV` | The command is part of the TPM's EA functionality and is used to integrate the content of an NV Index into a policy (cf. [117], Section 23.9). The command also allows to use a comparison operation, e.g., to check if the Index has some predefined value. | This command is used, e.g., to monitor BMS replacements. |
| `TPM2_PolicySigned` | The command is part of the TPM's EA functionality and is used to request an authorization before the associated command can be executed, e.g., a key can be used (cf. [117], Section 23.3). A public key that needs to be used for the respective signature is registered with the policy. | The command is used to verify the presence of the approved BMS within the `secEVCS`. |
| `TPM2_PolicyTicket` | The command is part of the TPM's EA functionality and uses as input a ticket produced by `TPM2_PolicySigned` after validating an authorization. The authorization ticket has an expiration time showing how long the particular authorization is valid (cf. [117], Section 23.5). It allows to reduce amount of the performed checks, if it can be assumed that the results hold for a certain period of time. | This command is used for performance optimizations in the `secEVCS`. |

| Command | General usage | Usage in thesis |
|---|---|---|
| TPM2_PolicyPCR | This command is part of the TPM's EA functionality and is used to verify that the contents of selected TPM's PCRs have the expected value (cf. [117], Section 23.7). The command provides a deferred assertion, i.e., if the PCRs' value is changed during the policy session (after the value has been successfully verified), the respective authorization will become invalid (cf. [114], Section 19.7.7). | This policy is used to represent the integrity state of the Electric Vehicle Communication Controller (EVCC)'s firmware. |
| TPM2_EC_Ephemeral | The command is used to create an ephemeral Elliptic Curve (EC) keypair provided as an input to TPM2_ZGen_2Phase (cf. [117], Section 19.3). | The command is used, e.g., in the PnP protocol for Elliptic Curve Diffie-Hellman (ECDH) Key Exchange (KE). |
| TPM2_ZGen_2Phase | The command implements the second phase of the KE and is used in combination with TPM2_EC_Ephemeral (cf. [117], Section 14.7). | The command is used to derive the symmetric session key in the PnP protocol. |
| TPM2_EncryptDecrypt2 | The command is used for symmetric encryption and decryption with a provided key (cf. [117], Section 15.3). | The command is used, e.g., in the PnP protocol to decrypt/encrypt protocol data. |
| TPM2_Quote | The command is used as part of the attestation scheme to quote PCR values and returns the signed digest of the selected PCRs (cf. [117], Section 18.4). | The command is used in the PnP protocol to determine the EV state. |
| TPM2_PCR_Extend | The command is used to update the value of the selected PCR (cf. [117], Section 22.2) | This command is used in the PnP protocol to describe the changed EV state after the installation of the update. |

# Part II.

# Securing Vehicle-to-Grid Communications

# 3. Securing Value-Added Services for Electric Vehicle Charging

In this chapter, we investigate security of communication protocols in the Vehicle-to-Grid (V2G) infrastructure and secure service provisioning that employs the emerging connectivity between previously disjunct actors. We describe Plug-and-Patch Protocol (PnP) that enables secure update of electric vehicles during charging as a value-added service offered by a charge point supporting the communication standard ISO 15118. The results of this chapter have been published in the paper "Plug-and-Patch: Secure Value Added Services for Electric Vehicle Charging" [3].

One of the use cases realized by the V2G infrastructure is public charging of Electric Vehicles (EVs) often considered to be the key to success of e-mobility. According to various studies [118, 119], the visibility of public charging can animate drivers of conventional cars to move on to EVs. However, the development of public charging currently offers little incentives to prospective service providers due to high costs and low profitability and, thus, is strongly dependent on state subsidies. New business models around IT-based Value Added Services (VASs) offered via Charge Points to drivers while EV's battery is being topped up are seen as a possible solution to increase industry participation [118].

Emerging charging and billing processes already rely on IT to reduce operational costs. The Plug-and-Charge (PnC) feature introduced in the international standard ISO 15118 [37, 38] uses high-level communication (over powerline or WiFi) to enable automated contract-based charging of EVs. Thereby, Charge Points serve as communication gateways connecting EVs to the provider's backend system to authenticate customers and collect billing information. The standard also defines a basic service discovery for various VAS types, which can be used to add arbitrary functionality to the charging process. Yet, security of VAS is not specifically addressed. Considering that services such as the Internet service or parking status service specified in the upcoming edition ISO/FDIS 15118-20 [39] may process sensitive personal and safety-critical information, this needs to be improved.

As an exemplary service scenario, we consider secure update of EVs using PnC. Since modern vehicles can have up to 100 Electronic Control Units (ECUs) [120] running divers firmware that requires regular updates, such service is also critical for the overall security [59, 121]. While high-end cars can often execute updates over-the-air (OTA), less expensive vehicles may not have such means. As car recalls are expensive and time-consuming, manufacturers need an alternative way to distribute firmware patches for these vehicles. In case an EV supports PnC, we propose to use the connection to the charging infrastructure to update the vehicle during charging. Using such VAS to update EVs has several advantages. In order to avoid being stuck with the insufficient range, EV users tend to charge at every opportunity, meaning that updates can be delivered within a short time. Most of the EV's functionality is idle while charging, which allows most ECUs to be updated automatically. Considering that even fast charging takes up to 30 minutes [122], it should give enough time to receive all data. In contrast to updates over a proprietary telematics link of Original Equipment Manufacturer (OEM), a VAS also allows other vendors to reach EVs and thus can be offered as a bonus to a charging contract.

**Contributions.** We propose a secure communication protocol for remote automotive updates performed via the V2G infrastructure and provide a concept for integration of this protocol as a VAS into the existing communication framework.

In summary, we make the following contributions:

- We provide a comprehensive analysis of VASs for EV charging and develop an integration concept for ISO 15118 and OCPP 2.0 ensuring backward compatibility.
- We propose the Plug-and-Patch (PnP) protocol to enable secure and reliable delivery of updates from an OEM to an EV using the V2G infrastructure in face of untrusted intermediaries and powerful adversaries. The EV's software state can be verified before and after the installation of updates using remote attestation based on Trusted Platform Module (TPM) 2.0;
- We analyze performance tradeoffs of PnP service using the proof-of-concept implementation and provide recommendations for the prospective service providers based on these approximations.

This chapter is organized as follows: Section 3.1 presents our system model and provides some additional background on automotive update mechanisms and deployment of these mechanisms in the V2G infrastructure. In Section 3.2, we specify functional and security requirements for the proposed PnP service. The analysis of V2G protocols with regard to the VAS functionality follows in Section 3.3. The PnP protocol is introduced in Section 3.4. In Section 3.5, we describe how to integrate such services into PnC and analyze, whether the PnP service meets the specified requirements. The related work is presented in Section 3.6, and the concluding remarks are given in Section 3.7.

## 3.1. System Model and Use Case: Remote Software Update using V2G Infrastructure

In this section, we provide the background on remote software updates and present the PnC infrastructure and underlying communication standards used as a basis for the proposed PnP service.

### 3.1.1. Remote Software Update in Automotive

The purpose of software updates is to change and improve software on existing systems such as automotive ECUs. Updates can be deployed by manufacturers to fix software bugs, add or enhance available features, or modify system components. The frequency and the size of updates depend on the target component of the vehicle, e.g., telematics system updates need to be performed more often and are relatively large (up to several MB), while some simpler ECUs may never be updated at all. In case of safety-critical bugs in ECU software, it is critical for car makers to provide fixes as soon as possible, otherwise, they can be held liable for a potential accident.

Automotive updates can be performed at repair shops, which usually incurs high recall costs and inconvenience for car owners. Using Software-Over-the-Air (SOTA), the update file is delivered to the vehicle remotely over a mobile or WiFi network. SOTA can be pushed to a vehicle by a remote server, or actively requested (pulled) by the vehicle itself. In this work, the latter strategy is adopted, as the manufacturer generally cannot predict when an EV is charging and, thus, ready to receive an update.

The steps to remotely update an EV are as follows:

1. EV and OEM establish a connection;

2. EV requests an update;

3. OEM verifies if the EV is eligible to receive the update;

4. OEM sends the update to the EV;

5. EV verifies the update;

6. EV installs the update;

7. OEM verifies if the update is successfully installed.

For SOTA, steps 1, 2, and 4 are usually implemented over the proprietary telematics link. If an update provider is not the vehicle's manufacturer, an additional agreement needs to be in place to deliver the patches of this provider to this vehicle. Step 3 is important for Intellectual Property (IP) protection with regard to the update code. Step 5 protects the EV from installing an incompatible or corrupted update. The installation in step 6 is out of scope for this work. However, the respective techniques can be found in [123, 124, 125]. The final step allows OEMs to avoid liability issues if an accident's cause has been fixed by an issued update [120]. If a vehicle does not support SOTA over the telematics link, a network connection offered by the PnC infrastructure can be used instead. We elaborate on this opportunity in the next section.

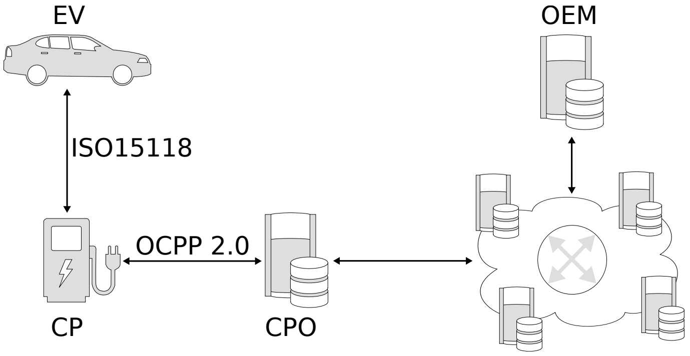


Figure 3.1: System Model

### 3.1.2. Software Update via PnC Infrastructure

In order to integrate the PnP service for EVs into the charging process, the capabilities of the underlying infrastructure need to be considered. Only a part of the overall V2G infrastructure is relevant to our scenario. Figure 3.1 shows actors, communication links, and protocols in the scope of our system model. We consider an EV charging at a Charge Point (CP) and wishing to check if an update is available for one of its software components. Since the vehicle cannot communicate with the respective OEM using wireless technologies like GSM, it attempts to communicate with its OEM by leveraging the VAS services offered by the CP. The CP is connected to the backend system of the Charge Point Operator (CPO) responsible for management of this Charge Point and its connection to the electric grid. The CPO may be the energy provider itself or a third party operating CPs. There is a service agreement between the CPO and the OEM according to which the CPO is able (and obliged) to relay messages it receives from the CP to the vehicle's OEM via the standard Internet connection.

In PnP, messages should propagate from the OEM to the EV and vice versa. Thus, a communication channel involving intermediate actors needs to be established, whereby the PnP messages are transferred using the communication protocols implemented between these actors. If any, the communication between the EV and Charge Point is provided using ISO 15118 [37, 38], while Charge Point and CPO usually communicate via Open Charge Point Protocol (OCPP) [40]. ISO 15118 features both service concepts PnC and VAS such as reservation of CPs or Internet access for infotainment during waiting time [37]. Though only service discovery is specified in ISO 15118-2, the VAS concept can be used to integrate the PnP related communication. Starting with the version 2.0 [83], OCPP offers support for the PnC service compliant with ISO 15118. Similarly, the task of OCPP 2.0 regarding the PnP service is to transport the respective VAS data between CP and CPO. The communication between the OEM and the CPO is currently not standardized, i.e., an arbitrary communication protocol can be deployed.

## 3.2. Requirements Analysis

In this section, we define functional and security requirements for the PnP system as a VAS for charging EVs. The considered adversary and threat model are described as well.

### 3.2.1. Functional Requirements

Outdated (vulnerable) software exposes EVs to security and safety risks that can lead to dangerous traffic situations. Following good practice to keep an EV's software up-to-date, the EV shall initiate the update service whenever it recharges and the VAS is available *(R1)* and install the patch as soon as possible *(R2)*. However, interleaving several updates is error-prone. The EV shall therefore finish any pending updates before requesting a new one *(R3)*. Furthermore, EVs shall never update components crucial to their current usage *(R4)* to prevent the vehicle from, e.g., becoming undrivable on the road [126]. Installing an incompatible update may also have severe consequences for the EV's safety. The vehicle shall be able to verify that a received update file matches the installed software and devices, i.e. its current state *(R5)*. As a failed software installation may actually brick devices [127], the EV shall be able to recover and continue its operation with the previous version if the update fails *(R6)*. Although this option may expose a vehicle to rollback attacks, it is necessary to ensure its safe operation. We consider the PnP service like any VAS to be secondary to the main functionality of EV charging. Thus, the handling of an update process shall not impact an ongoing charging process *(R7)*.

Several functional requirements are to be implemented by the OEM, Charge Point, and CPO. The OEM shall keep the update history for every produced EV *(R8)* to be able to determine whether the vehicle requesting an update is in a valid state. Moreover, the OEM shall only distribute updates that match vehicle's current state *(R9)*. The intermediary actors Charge Point and CPO shall forward any VAS data.

### 3.2.2. Threat Model

Our threat model assumes a powerful adversary, who has full control over the network connections between the EV and OEM (see Figure 3.1) and may read, re-route, insert, modify, or block transmitted messages. The adversary is only limited by the strength of the used encryption mechanisms [128]. Additionally, the adversary is able to manipulate the EV and Charge Point or replace their components as well as extract sensitive data *(T5)*, except in cases where the data is tamper protected, e.g., using an Hardware Security Module (HSM). These are realistic attack scenarios as was demonstrated by Miller and Valasek in the famous Jeep hack [129] and by Dahlheimer for the case of a Charge Point's firmware manipulation [56]. We assume that the OEM provides trustworthy (non-malicious) software updates. Any intermediate entities like the CPO or communication service providers are considered as potential adversaries.

With regard to the capabilities of the adversary, the following threats need to be considered for the PnP service. The adversary can tamper with any VAS related data *(T1)*, spoof the transferred messages *(T2)*, or replay previously captured ones *(T4)*. Adversaries may be able to learn and disclose sensitive information *(T3)*, e.g., the OEM's IP. If authentication is omitted in ISO 15118 (see Section 7.3 in [38]) and OCPP 2.0 (see Section A in [130]), the adversary can impersonate any of the actors in PnP *(T6)*. A lack of authentication also allows adversaries to repudiate their actions *(T7)*. A Man-in-the-Middle (MitM) attack *(T8)* also becomes possible without the proper authentication. Furthermore, as the PnP system deals with updating software running on a vehicle, the considered adversary also poses the threat of code replacement *(T9)*.

### 3.2.3. Security Requirements

In order to mitigate the threats defined in Section 3.2.2, we define the following security requirements. First, we require all underlying connections and protocols, i.e. ISO 15118, OCPP 2.0, and the connection between CPO and OEM, to be secured and authenticated using Transport Layer Security (TLS) *(S1)*. This already mitigates many network threats. Next, we require sensitive messages and data sent between OEM and EV to be encrypted end-to-end in addition to the TLS channels *(S2)*. This way, confidential information is protected against a rogue CP or CPO. Furthermore, to protect against the CP or CPO manipulating the encrypted data *(T1)*, we require data to be authenticated *(S3)*, e.g., using Message Authentication Codes (MACs) or digital signatures. The latter also helps enforce non-repudiation (countering *T7*). Using further authentication also protects against MitM *(T8)* and impersonation *(T6)*. Freshness of messages is also necessary to prevent an intermediary from replaying old messages or even an old update *(T4)*. Thus, we also require that all received messages shall be checked for their freshness *(S4)*. Authentication only works if the used credentials are trustworthy. Therefore, we require the PnP system to support the validation, revocation, and update of any authentication means *(S5)*. A physical adversary can simply extract credentials from the EV. Therefore, we require publicly accessible devices to store sensitive credentials in secure memory like HSMs *(S6)*. A TPM can additionally be used to prove the device's integrity based on the remote attestation.

## 3.3. Analysis of V2G Protocols

In order to implement PnP as a VAS, we need to make sure that the underlying infrastructure is able to support it. Thus, we analyzed the communication protocols ISO 15118 and OCPP 2.0 with regard to the functional and security requirements we defined for the PnP service in Section 3.2. This section presents the findings of the performed analyses.

### 3.3.1. Functionality

The protocols ISO 15118 and OCPP2.0 enable the communication between their respective peers EV ↔ CP and CP ↔ CPO. In OCPP 2.0, VAS can be realized via arbitrary data transfer (see Functional Block P - DataTransfer and Sec. 1.13

in [130]). However, *messageIDs* in *DataTransferRequest* message should be chosen carefully to ensure interoperability.

In contrast, ISO 15118 only supports VAS service discovery and does not specify a message sequence for VAS functionality. However, the *ServiceDiscoveryReq/Res* (Sec. 8.4.3.3 in [38]) and *ServiceDetailReq/Res* (Sec. 8.4.3.4 in [38]) messages can be used by EVs to check, whether a VAS with the given *ServiceID*, e.g., PnP is offered by the CP. ISO 15118's strict timing constraints (see Sec. 8.7 in [38]) may hinder service delivery, if it involves extensive communication with providers. Thus, VAS messages need to be integrated into the V2G charging loop.

### 3.3.2. Security

Our update system uses ISO 15118 and OCPP 2.0 for the transport of own data and relies on their properties for secure and reliable delivery (see *S1*). Both protocols offer secure channels based on TLS 1.2 for this purpose. However, the ISO 15118 specification [38] has two shortcomings: it uses outdated cipher suits with insecure CBC mode and unilaterally authenticated TLS channels, where the vehicle's identity is not verified. The vehicle authentication for PnC (using contract credentials) happens on the application layer, under the assumption of the trustworthy Charge Point and CPO and secure protocol tunneling. Therefore, VAS should only be offered after the EV's identity is verified.

In OCPP 2.0, TLS is optional and is used only with one security profile. We treat this profile as default for our system. Though TLS provides an authenticated exchange and prevents the replay of captured messages, it cannot prevent data manipulation in the systems of the intermediary actors.

The protocols provide certain protection for confidentiality (*S2*) and authenticity (*S3*). ISO 15118 applies encryption when updating or installing the vehicle's contract certificate (cf. Sec. 8.4.3.10 and 8.4.3.11 in [38]). In that case, symmetric encryption with a key derived using an ephemeral-static Diffie-Hellman key exchange is used (cf. Sec. 7.9.2.4.3 in [38]). The standard demands every V2G entity to support this key exchange in requirement [V2G2-122] [38]. ISO 15118 also applies an XML signature mechanism for some messages such as meter readings (see Table 13 in [38]). A VAS can reuse these mechanisms to ensure the compatibility with the systems of EV and Charge Point.

The standards also provide some level of protection against replay attacks (*S4*). Besides replay protection offered by TLS, ISO 15118 requires unique session IDs (see Sec. 8.4 in [38]). Re-sending messages in another session is thus not possible. Due to the strict sequencing and error handling of ISO 15118, replaying messages in the same session is also hard. In contrast, OCPP 2.0 demands session IDs only to be unique per sender (see Sec. 4.1.4 in [84]), which makes replaying messages to a different recipient possible. Thus, we require IDs to be universally unique.

ISO 15118 and OCPP 2.0 support validation and revocation of X.509v3 public key certificates. A certificate is validated using Certificate Revocation Lists (CRLs) and Online Certificate Status Protocol (OCSP) (see Sec. 7.3.2 in [38] and Sec. A in [130]) according to RFC 5280. Still, OCPP 2.0 uses fast expiry for the CPO's certificate (A00.FR.701 in [130]) instead. Since ISO 15118 requires the CP to cache OCSP responses, online certificate validation may be used, too. Certificate update methods are described in Sections A02 and A03 for OCPP 2.0 [130] and in Section 8.4.3.10 for ISO 15118 (contract certificate) [38].

Neither OCPP 2.0 nor ISO 15118 consider integration of HSMs to protect cryptographic keys and provide a secure environment for cryptographic operations. Both standards focus on the definition of communication protocols but leave their secure implementation out of scope. We see the usage of an HSM, such as a TPM, necessary to secure the cyber-physical components of the charging infrastructure such as EV and CP. We thus propose the explicit use of HSMs to be considered in future versions of these standards.

## 3.4. Plug-and-Patch System

In this section we present our Plug-and-Patch (PnP) system that allows EVs to remotely receive updates from their OEMs using a VAS during charging. The proposed solution consists of protocols and algorithms running on the EV, CP, and backend systems of CPO and OEM. PnP follows the procedure for automotive updates discussed in Section 3.1.1. The session establishment builds on the basic SIGMA protocol [131] to enable an authenticated and secure communication between the EV and the OEM over the charging infrastructure. In order to verify the vehicle's state before and after the update and confirm its installation PnP uses attestation in form of TPM2_Quotes (see Section 9.5.3 in [114]).

The system works under realistic assumptions and requires several preliminary steps as described below. The regular flow of the PnP protocol is described in Section 3.4.2. However, errors may occur in PnP, e.g., when the update has to be deferred. These error and edge cases are described in Section 3.4.3. The protocol's message flow can be integrated into the ISO 15118 charging loop with minimal effort as shown in the proof-of-concept evaluation.

### 3.4.1. Assumptions and Preliminaries

Below we describe the assumptions with regard to the PnP protocol and its environment as well as necessary preliminaries.

**PnP Identity Credentials.** We introduce a new set of keypairs and X.509v3 certificates to be used with the PnP service. This way, the purpose of the existing ISO 15118 certificates is kept distinct and an update cannot be related to the EV user, which is important for privacy reasons. The *Vehicle Update Certificate (VUC)* and associated keypair $\{EV_{pub}, EV_{sec}\}$ is used to uniquely identify the vehicle. Similarly, the *OEM Update Certificate (OUC)* with $\{OEM_{pub}, OEM_{sec}\}$ serve the same purposes for the OEM. Both of these credentials are generated by the OEM and, in case of the VUC and its keys, are installed in the EV during manufacturing. We also assume that the OUC is present on the EV before the first execution of the update protocol. Additionally, the certificates are part of the OEM Root CA as defined in ISO 15118 (see Figure E.1 in [38]). We assume that all certificates and keypairs can be updated and validated using, e.g., OCSP or CRLs.

**Certificate Validation.** PnP uses the OCSP to validate the EV's or OEM's certificate. The OEM can check the VUC's status by simply querying a trusted OCSP responder, while the EV may not have such capability. We hence let the OEM retrieve OCSP responses for the OUC. To counter replay attacks, the EV sends the OEM a nonce that is used to retrieve the response. The EV should also possess the OCSP responder's public key to be able to verify the authenticity of the response. In the message flow description, this is referred to by validating the OUC or VUC. The full certificate chains should be checked if the certificates are not directly signed by the OEM Root CA.

**TPM 2.0 and its Onboarding.** Our update system strongly relies on the functionality provided by a TPM 2.0. As TPMs are nowadays often present in vehicles [132], this is not restrictive. Within the TPM resides an asymmetric Attestation Identity Key (AIK) [133] created, installed, and bound to the TPM by the OEM during manufacturing. An AIK is bound to the TPM through its derivation of the Endorsement Key (EK), which is unique to a TPM. The AIK keypair serves the purpose of binding system attestation data and validating such bindings, e.g., a TPM2_Quote, to the vehicle. This is necessary to make sure that a vehicle gets exactly the update it needs and the correct states are reported and stored by the OEM. Binding the EV to its TPM is, e.g., done by storing the VUC and public AIK together. For example, a unique Vehicle Identification Number (VIN) could be used as a common database key. For this to work, it is required that during manufacturing of the EV, the OEM has verified the identity of the TPM. This can be achieved by running a protocol between OEM backend and TPM that generates an authentic and encrypted session and thus verifies that the TPM has access to the corresponding private key of the EK. An example of such protocol is provided in the RazorClamMCU project by Stefan Thom [134]. TPM2_Quote responses are bound to the TPM through signing them with the secret part of the AIK. An update session is bound to the EV as defined in the SIGMA protocol [131] during session establishment. SIGMA is also the basis for a secure and authenticated session establishment in our protocol.

**Setup.** We assume that the EV is in the V2G Charging Loop and selected VAS (see Sec. 8.7.4 in [38]). Furthermore, the OCPP 2.0 session is online and all intermediary actors are ready to relay VAS data. ISO 15118 and OCPP 2.0 are assumed to use TLS. Also, we assume that OEM and CPO backends are able to communicate securely on demand.

The OEM has the data about the EV's update history and the last known vehicle's state, i.e., the installed firmware. This data is required to decide if an update is available. Since TPM2_Quote does not contain any data over which the quote has been computed (see Sec. 17.6.2 in [114]), the OEM also has to have a database of known vehicle states and corresponding TPM2_Quote responses. Storing TPM2_Quote responses and vehicles states could be beneficiary for the OEM for the case of liability issues after accidents. Due to the provided non-repudiation, the OEM could prove that the EV had installed the expected patch. Moreover, providing an update file to a modified firmware can pose a risk to the OEM's IP by leaking the software package.

**TPM Primitives.** We assume the EV utilizes a TPM to securely store keys and provide a secure execution environment for cryptographic operations, i.e., signing, signature verification, en- and decryption, and ECDH key exchange.

### 3.4.2. Regular Message Flow

This section describes the regular flow of the PnP protocol. All messages consist of a header and a body. The header includes a globally unique sessionID chosen by the OEM, the messageID, and an integrity tag. This tag is either a digital signature or MAC computed over the contents of the message body, sessionID, and messageID. For brevity, only the message body and integrity tag are shown in the following descriptions.

**Session Establishment.** The first phase is to establish a secure and authenticated session between the EV and OEM (see Figure 3.2). This is done using an ephemeral Elliptic Curve Diffie-Hellman (ECDH) Key Exchange (KE) based on the basic SIGMA protocol (cf. Sec. 5.1 in [131]). In step (1), the vehicle first generates its ephemeral keypair and two nonces, one for the ECDH and the other for validating the OUC with OCSP. A *DHKeyExchangeRequest* message ($DHReq$) including the VUC, ephemeral public key, and the nonces is then sent to the CP. In turn, the CP forwards the message to the CPO, who deduces the EV's manufacturer from the VUC and connects to the respective OEM using a regular TLS channel. Notably, the CP and the CPO cannot manipulate PnP messages due to the integrity tag. For brevity, the forwarding of messages by the CP and CPO is omitted from the remaining description.

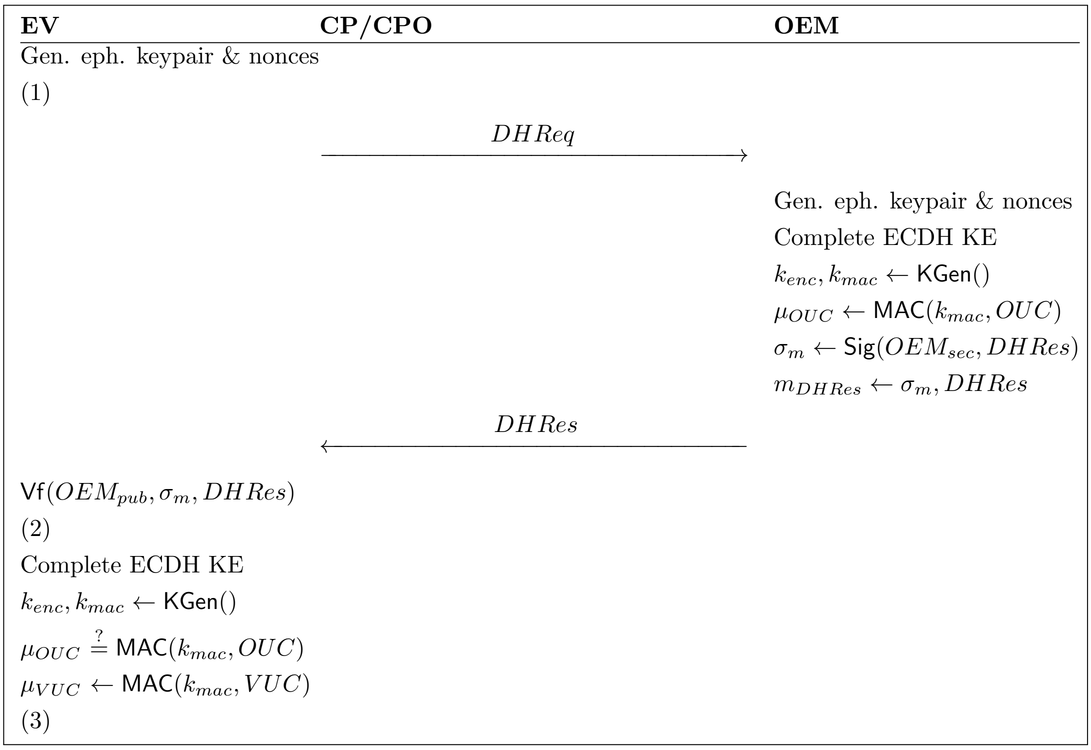
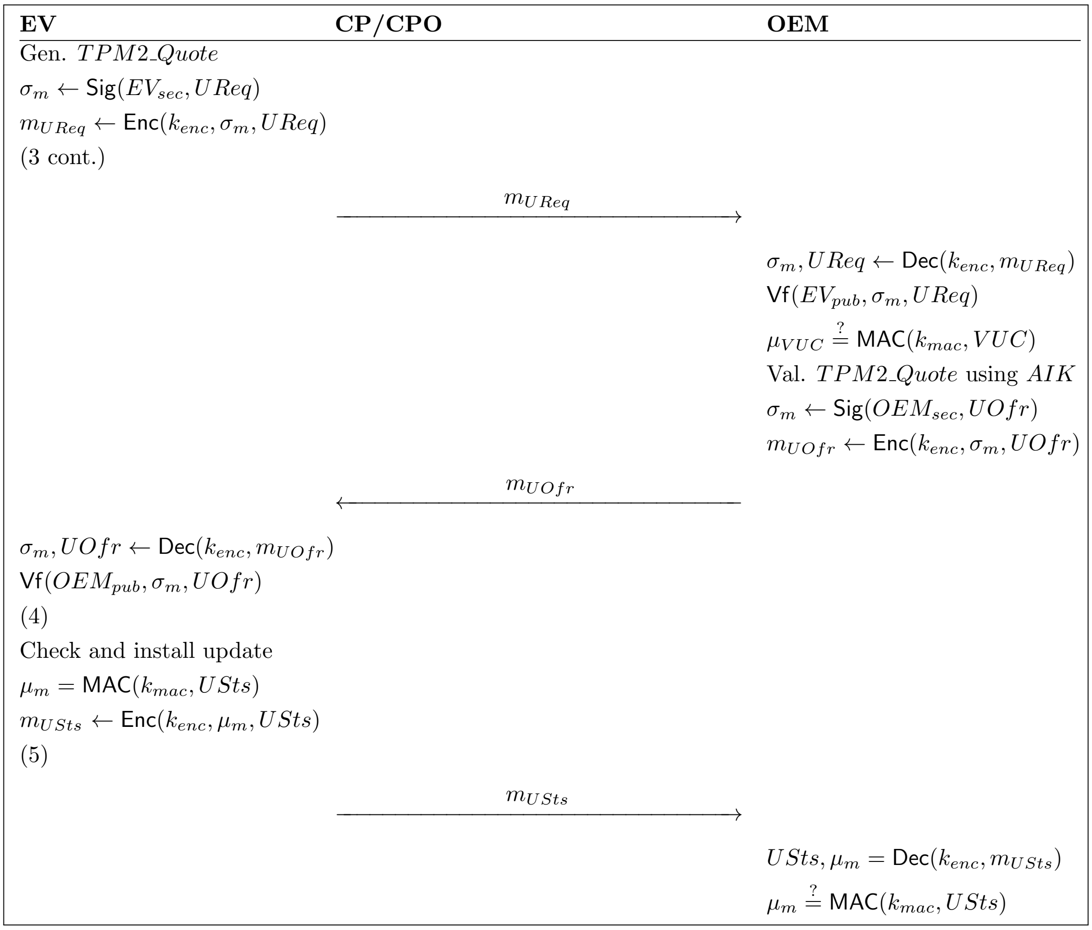

EV
CP/CPO
OEM
Gen. eph. keypair & nonces
(1)
$DHReq$
Gen. eph. keypair & nonces
Complete ECDH KE
$k_{enc}, k_{mac} \leftarrow \mathsf{KGen}()$
$\mu_{OUC} \leftarrow \mathsf{MAC}(k_{mac}, OUC)$
$\sigma_m \leftarrow \mathsf{Sig}(OEM_{sec}, DHRes)$
$m_{DHRes} \leftarrow \sigma_m, DHRes$
$DHRes$
$\mathsf{Vf}(OEM_{pub}, \sigma_m, DHRes)$
(2)
Complete ECDH KE
$k_{enc}, k_{mac} \leftarrow \mathsf{KGen}()$
$\mu_{OUC} \stackrel{?}{=} \mathsf{MAC}(k_{mac}, OUC)$
$\mu_{VUC} \leftarrow \mathsf{MAC}(k_{mac}, VUC)$
(3)

Figure 3.2: Secure Session Establishment

In step (2), the OEM first validates the VUC. Next, the OEM also generates an ephemeral keypair and two nonces, one for the ECDH and the other for a subsequent TPM attestation, before deriving the shared secret using the ECDH nonces and the ephemeral keys. From the secret, two keys are derived using the key generation function `keyGen()`: the symmetric session key $k_{enc}$ and the symmetric key for MACs $k_{mac}$. An OCSP response for the OUC is retrieved using the OCSP nonce provided by the vehicle. A MAC over the OUC $\mu_{OUC}$ is computed to bind the OEM's ID to the session with the session's specific $k_{mac}$. The response *DHKeyExchangeResponse* message (DHRes) containing $\mu_{OUC}$, the nonces, the OEM's ephemeral public key, the OUC, and the OCSP response is signed and is sent back to the EV.

In step (3), the EV validates the OUC using the OCSP response and its own nonce and verifies the OEM's signature on the received $DHRes$ message before completing the ECDH on its side. Then, the shared secret and symmetric keys are computed. Using $k_{mac}$, $\mu_{OUC}$ is validated and $\mu_{VUC} = MAC(k_{mac}, VUC)$ is computed, as defined in SIGMA [131]. At this point, the secure channel between OEM and EV is established, the OEM and EV identities are confirmed, and the actual update procedure can start (see Figure 3.3).

**Checking, Executing, and Notifying the Update.** Figure 3.3 shows the main update procedure, i.e., checking, executing and notifying the installation of an update. In order to prove its state to the OEM, the vehicle executes a TPM2_Quote using the challenge supplied by the OEM. Finally, $\mu_{VUC}$ and the TPM2_Quote response are put in an *UpdateRequest* message ($UReq$), which is signed by the EV and encrypted using the session key $k_{enc}$ before being sent to the OEM.



EV
CP/CPO
OEM
Gen. $TPM2\_Quote$
$\sigma_m \leftarrow \mathsf{Sig}(EV_{sec}, UReq)$
$m_{UReq} \leftarrow \mathsf{Enc}(k_{enc}, \sigma_m, UReq)$
(3 cont.)
$m_{UReq}$
$\sigma_m, UReq \leftarrow \mathsf{Dec}(k_{enc}, m_{UReq})$
$\mathsf{Vf}(EV_{pub}, \sigma_m, UReq)$
$\mu_{VUC} \overset{?}{=} \mathsf{MAC}(k_{mac}, VUC)$
Val. $TPM2\_Quote$ using $AIK$
$\sigma_m \leftarrow \mathsf{Sig}(OEM_{sec}, UOfr)$
$m_{UOfr} \leftarrow \mathsf{Enc}(k_{enc}, \sigma_m, UOfr)$
$m_{UOfr}$
$\sigma_m, UOfr \leftarrow \mathsf{Dec}(k_{enc}, m_{UOfr})$
$\mathsf{Vf}(OEM_{pub}, \sigma_m, UOfr)$
(4)
Check and install update
$\mu_m = \mathsf{MAC}(k_{mac}, USts)$
$m_{USts} \leftarrow \mathsf{Enc}(k_{enc}, \mu_m, USts)$
(5)
$m_{USts}$
$USts, \mu_m = \mathsf{Dec}(k_{enc}, m_{USts})$
$\mu_m \overset{?}{=} \mathsf{MAC}(k_{mac}, USts)$

Figure 3.3: Main Update Procedure

After decrypting the update request message as well as verifying its signature, the OEM validates the TPM2_Quote response using the vehicle's public AIK and the nonce it generated for the attestation (step (3)). If the Quote is valid, the manufacturer is able to derive the EV's state based on the attestation data contained in the TPM2_Quote response and the according vehicle states from the database. The OEM can then deliver the required update to the EV.

In order to provide the update to the EV, the OEM packs the update and some metadata in an *UpdateOffer* message ($UOfr$). Alternatively, some information about why no update can be offered is put into the message. The OEM now signs the message so that the EV can install the update later after the session has ended. Furthermore, signing the update message makes sending this specific file non-reputable for the OEM. Encryption is applied before sending $m_{UOfr}$, containing the signature and offer message.

When receiving the update offer in step (4), the EV first decrypts the message and validates the signature. If no update was sent, the process ends and the session is terminated. Otherwise, the EV checks the received update file. We implement this with the help of metadata. This metadata could, e.g., describe for which vehicle and vehicle state the update is intended for. Following the successful check is the installation of the update package, which is out of scope for this work. After the installation, the EV sends a status update message to the OEM in step (5), reporting the successful installation. This *UpdateStatus* message ($USts$) is MACed and encrypted as $m_{USts}$, containing the MAC and status message, before sending.

**Proving the Update.** The final phase of the protocol serves the purpose of proving the update installation due to the liability reasons (Figure 3.4). For that, a TPM2_Quote command is executed again. After receiving, decrypting, and validating the EV's status message, the OEM generates another attestation challenge in step (6). The challenge is then

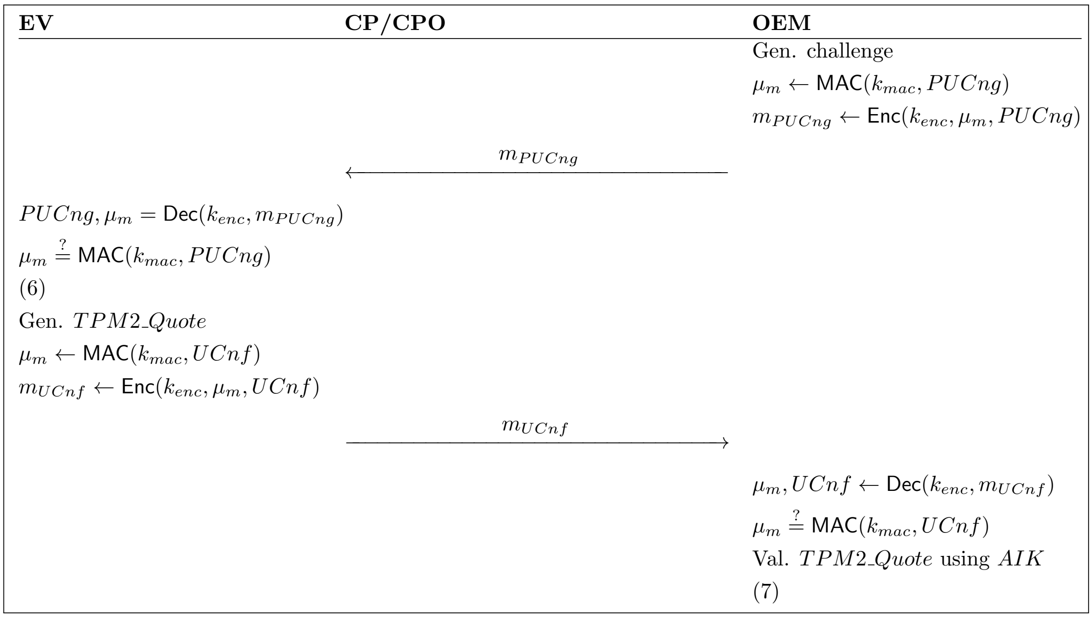

EV
CP/CPO
OEM
Gen. challenge
$\mu_m \leftarrow \mathsf{MAC}(k_{mac}, PUCng)$
$m_{PUCng} \leftarrow \mathsf{Enc}(k_{enc}, \mu_m, PUCng)$
$m_{PUCng}$
$PUCng, \mu_m = \mathsf{Dec}(k_{enc}, m_{PUCng})$
$\mu_m \overset{?}{=} \mathsf{MAC}(k_{mac}, PUCng)$
(6)
Gen. $TPM2\_Quote$
$\mu_m \leftarrow \mathsf{MAC}(k_{mac}, UCnf)$
$m_{UCnf} \leftarrow \mathsf{Enc}(k_{enc}, \mu_m, UCnf)$
$m_{UCnf}$
$\mu_m, UCnf \leftarrow \mathsf{Dec}(k_{enc}, m_{UCnf})$
$\mu_m \overset{?}{=} \mathsf{MAC}(k_{mac}, UCnf)$
Val. $TPM2\_Quote$ using $AIK$
(7)

Figure 3.4: Proving the Update to OEM

packed in a *PostUpdateChallenge* message ($PUCng$), which is also MACed and encrypted by the OEM before being sent as $m_{PUCng}$ to the EV. The message is decrypted and the MAC validated by the EV after receiving it. The EV uses the provided nonce for a TPM2_Quote and send the resulting response as an *UpdateConfirmation* message ($UCnf$). Again, the message is MACed and encrypted before sending it as $m_{UCnf}$ to the OEM. The OEM's final task in step (7) is to validate the quote against the AIK and a known, expected 'up-to-date' state after decrypting and validating the message. Now, the OEM has a proof that the update was indeed installed and the protocol is concluded. With that, the session between OEM and EV ends and so does the TLS session between OEM and CPO. The handling of the VAS may also be ended in ISO 15118, but the charging session and all relevant communication via ISO 15118 and OCPP 2.0 continues.

### 3.4.3. Error and Edge Cases

There are two possible cases, where the EV disconnects during protocol execution. If the EV disconnects before the update is offered by the OEM, the protocol simply terminates. Otherwise, the OEM notes that an update was sent and expects it to be installed before the EV requests an update again. When the EV requests a new update later, the OEM can determine the vehicle's state using the information from the *UpdateRequest*. This should either match the new updated firmware state or the previous state, thus, the EV may either be up-to-date or will have to repeat the update.

In order to register errors within our firmware update system, we prepare the *UpdateError* message containing an ErrorID and the *UpdateStatus* message with a StatusID. However, actual error handling, e.g., for installation failure, invalid MAC, signature or Quote, as well as general processing errors is considered out of scope. We consider it to be the task of future specifications to avoid security issues and compatibility problems between different vendors.

## 3.5. Evaluation

In this section, we evaluate the proposed PnP system using a proof-of-concept implementation, in order to determine, whether are satisfied by our solution satisfies the functional and security requirements from Section 3.2.

### 3.5.1. Implementation and Test-bed Setup

The proposed PnP system was implemented by extending the available Java implementations of ISO 15118[1] and OCPP 2.0 with the PnP functions described in Section 3.4. For the TPM functionality, Microsoft Research's TSS 2.0[2] and IBM's TPM simulator[3] were used. For certificate validation and OCSP responses retrieval, the BouncyCastle[4] Java API was employed. The cryptographic algorithms AES-256-CFB for symmetric encryption, SHA256 as a hash function, and NIST P-256 curve based ECDH were chosen for the evaluation in accordance with the standards' specifications.

In the PnP prototype, the ISO 15118's messages *ChargingStatusRes* (Sec. 8.4.4.2 in [38]) and *CurrentDemandRes* (Sec. 8.4.5.4 in [38]) were extended with a *VASDataAvailable* flag to indicate that VAS data are ready to be retrieved by the EV. Additional messages *VASDataReq* and *VASDataRes* were introduced to carry the PnP data. Both messages contain a *ServiceID* reserved for PnP and a *VASData* field with the actual data. The response message also includes the *ResponseCode*, as demanded by ISO 15118, and the *VASErrorID* and *VASErrorMessage* for error handling.

The test-bed setup consisted of two Raspberry Pis 3 B+ (Broadcom BCM2837B0, 1.4 GHz, 1GB RAM) as the EV and the CP, and an Asus UX305F Laptop (Intel Core M 5Y10, 8 GB RAM) for the backend services (CPO, OEM, OCSP). The Pis were connected directly via Ethernet and the Pi running the CP was connected to the Laptop via WiFi. The regular PnP protocol flow and edge cases were tested with different update sizes. Also, performance measurements were carried out to determine the overhead of the proposed solution with regard to execution times (computational overhead) and network traffic (communication overhead). The measurements were performed with the following parameters: a charging session lasted 4000 cycles of the V2G charging loop and a raw update file was 128 KiB in size. Table 3.1 lists the mean values derived from a series of 100 experiments.

Table 3.1: PnP Protocol Performance

| Parameter | w/o PnP | w/ PnP | Overhead |
|---|---|---|---|
| Computational | 273.39s | 293.96s | 19.58s |
| Communication | 3923.95 KiB | 4175.19 KiB | 251.23 KiB |

### 3.5.2. Findings and Discussion

The evaluation showed that a large update file may lead to a session termination due to the V2G_SECC_Sequence_Timeout (cf. Table 109 [38]) in ISO 15118. When 60s timeout is reached, the CP terminates the V2G session. Our test system reached this timeout when decrypting update files larger than 512 KiB. Therefore, large VAS processing tasks need to be handled outside the V2G charging loop. Large data packages may also have to be split up, because of the available PLC bandwidth, low backend connectivity via cellular networks, limited EV/CP storage, and the duration of the charging session. As PLC can achieve transfer rates of hundreds of MBit/s [135, 136], its effects are negligible. Delta update schemes [137], where only the changed code is flashed, can help to resolve the mentioned limitations.

As Table 3.1 shows, executing the PnP protocol has an overhead of 19.58s. Considering that decrypting the update file takes 17.06s and is currently realized within the charging loop, the overhead of the protocol itself drops to 2.51s. In terms of network traffic, PnP adds 251.23 KiB overhead. However, the message containing the update is 227.97 KiB (due to encryption and JSON/Base64 encoding), constituting by far the largest amount of data in our protocol. The evaluation, therefore, shows that PnP adds little overhead to a charging session. Also, we show that most of the overhead depends on the update size, further motivating smaller update file sizes.

Since various events can affect the service delivery, error handling is crucial, especially, for paid services to ensure that users are not billed without actually receiving the service. The successful consumption of PnP, and any other VAS, depends on the following events: (i) the PnC session start, i.e., an EV needs an opportunity to request the VAS, (ii) potential errors that may impact the VAS handling, e.g., ISO 15118 errors that result in the session and/or VAS termination, and (iii) the actual time the ISO 15118 connection is available, i.e., how long the EV takes to charge. The duration can be estimated based on the charging parameters (e.g., amount of energy, departure time).

[1] https://github.com/V2GClarity/RISE-V2G
[2] https://github.com/microsoft/tss.msr
[3] https://sourceforge.net/projects/ibmswtpm2/
[4] https://www.bouncycastle.org

### 3.5.3. Implementation of Requirements

In this section, we discuss how PnP implements the functional and security requirements defined in the Section 3.2.1 and Section 3.2.3, respectively.

*(R1)* is realized as the VAS can be selected from the offered services in our ISO 15118 implementation. However, an option is available to simulate that an update is still pending to be installed, resulting in not requesting the VAS, thus, fulfilling *(R3)*. As the actual update installation is out of scope of this work, *(R2)*, *(R4)*, and *(R6)* are currently not implemented. PnP fulfills *(R5)* by attaching metadata to the update file. This extra data represent information about the update and is checked against the vehicle's state upon reception. This could also be used to implement *(R4)*. As the integration of VAS messages in ISO 15118 is done in a way, where potential errors do not affect the rest of the charging process, *(R7)* is met. The functional requirements *(R8)* and *(R9)* are realized in the following way: the implementation of the OEM backend includes a small database that stores the EV's last known state as well as possible known attestation results. Based on the received attestation data within a TPM2_Quote response, the OEM is able to derive the EV's current state and, thus, assemble the correct update package.

As for the security requirements, *(S1)* is fulfilled by using a TLS connection between the OEM and CPO backends. Furthermore, ISO 15118 and OCPP 2.0 are also only used in their TLS modes. As described in Section 3.4.2, all data transferred between EV and OEM are encrypted using the established session key. Thus, *(S2)* is also met. Furthermore, all data is either digitally signed or MACed by the sender, implementing *(S3)*. Freshness of all messages, required in *(S4)*, is achieved with a unique session ID for each run of our update protocol. Our system uses OCSP to validate certificates, hence, implementing *(S5)*. ISO 15118 also defines how credentials could be updated (see Sec. 8.4.3.10 in [38]). This mechanism could thus be used as a base to update our system's credentials. Lastly, our protocol relies on TPM 2.0 for system attestation. Therefore, we can also use it to protect sensitive data and credentials as demanded by *(S6)*.

## 3.6. Related Work

Significant effort has already been put into developing secure SOTA methods. In this section, we present several such systems next to our approach. The work [138] presents a protocol for secure Firmware Over-the-Air (FOTA) updates aiming at vehicle's ECUs with limited resources. The protocol achieves confidentiality, integrity, and authenticity for the update in transit. However, this is done using pre-shared symmetric encryption keys, instead of unique session keys. The authors address this problem in [139] by renewing the keys periodically. Furthermore, their system does not authenticate the user. In [140], the authors address verifying the installation through checking an ECU's memory, while PnP employs the platform attestation. Similarly, the work [141] uses symmetric encryption with one-time session keys for confidentiality protection and pre-shared keys for vehicle authentication. An update's integrity is verified using message digests. However, update installation is not considered. In [142] the author specifies a high-level scheme, where a vehicle can request an update from the manufacturer like in our approach. Yet, security details are not specified.

The EVITA project proposed a system for OTA updates [143], which integrates HSMs similar to PnP. An authenticated key exchange is used to establish a symmetric session key. Moreover, an update's installation is verified by testing the values of ECU Control Registers. Subsequent analysis of this update scheme in [127] showed that bricking the updating ECUs was possible and, therefore, the EVITA+ protocol was proposed to address this issue. An OTA firmware update scheme using TPM 2.0 was described in [120]. The authors use TPM2_Quotes to check the vehicle's state before and after the update. The latter check is sent to the backend as a proof. Also, the TPM is used to check the update's authenticity and is suggested for session establishment. Thus, this system shares similarities with PnP.

Security analyses of ISO 15118 and OCPP were also published. Analyses of OCPP 1.6 were conducted in [94] and [95]. As these works analyze an older version of the protocol, some of the flaws were fixed in the edition OCPP 2.0. The ISO 15118 analysis in [50] shows gaps specific to the charging process. Thus, these analyses supplement our work.

## 3.7. Chapter Summary

In this chapter, we describe Plug-and-Patch, a secure and reliable update system for EVs leveraging capabilities of the V2G infrastructure. Moreover, we provide the first, to the best of our knowledge, full-fledged definition of Value Added Services in the context of the standards ISO 15118 and OCPP2.0. PnP also integrates TPM 2.0 as an important building block to verify an EV's software state before and after installation of updates using attestation. The proposed design builds on the charging communication standards, relies only on already available connections, does not oblige CPs to support any proprietary service implementations, and can solve the OEM lock-in for SOTA. Our proof of concept implementation and evaluation of the PnP service shows that updates via the charging infrastructure add minimal overhead to its core functionality. Due to the generic nature of our solution, PnP can be used to implement any arbitrary vehicle-related VAS, e.g., collecting vehicle's telemetry for predictive maintenance. The only disadvantage of the overall approach is the potentially limited update size due to physical constraints of the involved systems such as the low connection bandwidth and EV/CP storage capacity. This limitation can be resolved by using incremental update strategies.

# 4. Fraud Detection in Distributed Payment Systems

In this chapter, we investigate fraud detection methods, which can be employed to identify misbehaving Electric Vehicles (EVs) based on the traces of Vehicle-to-Vehicle (V2V) and Vehicle-to-Grid (V2G) transactions accumulated in the backend systems of service providers. As an exemplary service, we use the Mobile Money Transfer (MMT) service, representing one of the possible payment options in e-mobility and V2G services. In this case, EV users and V2G service providers are normal clients of the MMT service who can participate in diverse financial transactions. This chapter is based on on the previously published paper "No Smurfs: Revealing Fraud Chains in Mobile Money Transfers" [5].

MMT services enable funds transfer made directly on mobile devices of end-users using digital equivalent of cash (electronic money) without involving bank accounts. This also includes local and remote mobile payments to purchase goods and services, e.g., the Plug-and-Charge (PnC) service for EV battery recharge. MMT systems are subject to the same controls as those required for financial institutions, including the detection of *Money Laundering (ML)*. The risk of ML in MMT services raises serious concerns, because with digital currencies it is possible to transfer funds worldwide and avoid supervision of the regulators [144]. The goal of ML is to disguise the origin of illegal incomes and make them appear legitimate using a range of strategies to evade Anti-Money Laundering (AML) controls. We focus on an often practiced ML technique known as micro-structuring of funds or *smurfing*, which involves multiple third parties, so-called "smurfs", conducting money transfers on behalf of fraudsters, so that the transaction amounts are kept below reporting levels [145, 146]. A smurf, or more commonly termed money mule, is recruited by fraudsters as a financial intermediary, who accepts money from one fraudster and forwards it to another fraudster for a fee. Mules can be acquired through the use of phishing strategies, such as bogus jobs (e.g., financial manager or agent), and not always aware that they are dragged into illegal activities [147, 148]. Participating in ML, even unknowingly, is a criminal offense, and can have legal consequences for mules ranging from a frozen bank account, to legal prosecution and accountability for losses born by other victims of the fraud [149]. If proper controls are not deployed, fraudsters can get access to the service without disclosing their identity to the Mobile Network Operator (MNO) providing the service, for example, by taking advantage of prepaid phones, "pooling" or delegation of mobile devices [146]. Due to AML regulations valid in most countries, it is compulsory for MMT service providers to report ML activities. Therefore, reliable ML detection is critical for MNOs to be able to run mobile financial services and prevent reputation risks.

The common approach to fraud detection in MMT is to use classical statistical methods including machine learning and data mining [150, 151, 152, 153]. However, these methods need a training database, which can be difficult to obtain for ML transactions, and often produce results that are not easy to interpret. Another challenge for ML detection is that fraudulent transactions may have parameters (e.g., money amount, frequency) very close to regular money transfers and be nearly indistinguishable from the behavior of legitimate service subscribers.

**Contributions.** We propose a new method for ML detection in MMT services, Fraud Chain Detection (FCD), that is able to identify fraudsters and money mules engaged in a *fraud chain*, revealing the structure of the organized group. In contrast to the classical approaches, the FCD does not depend on any prior information about fraud patterns or samples of ML transactions, since it builds on a model-based approach for event-driven process analysis using only process specifications as input. Essentially, our method is an extension to our previous work Predictive Security Analysis at Runtime (PSA@R) [15]. While PSA@R provides a general framework for process analysis, the FCD defines specific usage behavior patterns related to ML activities in MMT and enables the identification of fraudsters in an MMT service by monitoring behavior of the end-users and matching it against processes in our model based on pre-defined events.

Since no real-world data were publicly available for our scenario at the time of publication, we evaluate the proposed solution on simulated transaction logs produced by an advanced MMT simulator based on a multi-agent platform [65]. Using synthetic data for testing offers several advantages, if the simulator allows generating data with realistic properties [154]. In our case, the lack of ground truth in real-world data would hinder the definition of detection errors, calculation of precision and recall regarding fraudulent transactions, and thus comparison of different approaches. We use the same logs for the comparative study of several classical machine learning algorithms to obtain comparable results required to judge about the efficiency of the new detection method. For evaluation purposes, we implement our method as a plug-in to an in-house process modeling and verification tool Predictive Security Analyzer (PSA) [34]. The FCD achieves better recognition performance in comparison with the standard techniques.

The remainder of this chapter is organized as follows: Section 4.1 presents the MMT ecosystem and defines the exemplary fraud scenario. In Section 4.2, we provide some additional background on fraud detection. In Section 4.3, we formulate the design goals and introduce our alternative method for detection of fraud chains. Results of our experiments and their comparison with machine learning algorithm are reported in Section 4.4. Finally, Section 4.5 concludes this chapter and provides some ideas for future research.

## 4.1. Fraud Chains in Mobile Payments

### 4.1.1. MMT Ecosystem

An MMT service is a complex ecosystem that involves an MNO, a private bank, the country's Central Bank in which the service is deployed, and service subscribers. An MNO provides infrastructure and communication services and in partnership with a private bank emits electronic money called *mMoney* [155]. mMoney is a digital equivalent of funds (cash) that can be used solely within the service system to conduct mobile-enabled financial operations, such as Airtime Recharge (AR), Money Deposit (MD) and Money Withdrawal (MW), Merchant Payment (MP), domestic and international Client-to-Client transfer (c2c). End-users and retailers are service subscribers, holding a prepaid mobile account *mWallet* stored on an MMT platform. As shown in Figure 4.1, in order to conduct c2c transfer, end-user Alice needs to convert her cash to mMoney and deposit its amount into her mWallet with the help of the retailer $R1$. Then, Alice can use her mobile device to transfer mMoney to Bob, if he is subscribed to the same MMT service. On receiving the transfer, Bob can withdraw cash from his mWallet at the retailer $R2$.

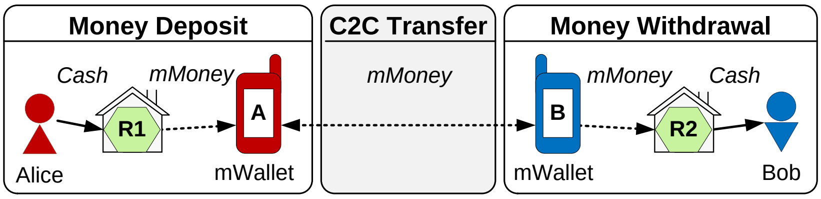


Figure 4.1: C2C Funds Transfer in MMT Service

### 4.1.2. Fraud Scenario

The fraud scenario studied in this work originates from an often practiced ML technique known as micro-structuring of funds or *smurfing*. Smurfing involves multiple third parties or third party accounts that conduct money transfers on behalf of fraudsters so that large amounts of "dirty" money are distributed among a number of smaller transactions [145]. It allows fraudsters to hide ML activities from controllers and evade AML reporting requirements, thus, reducing the likelihood of fraud detection.

**Definition 1 (Fraud chain)** *A* fraud chain *is a group of end-users of an MMT service identified by their mobile accounts that misuse the service to hide or disguise the origin of funds and to evade monetary record keeping and AML report requirements implemented by an MMT service provider to control mMoney transfers. The fraud chain consists of a sending fraudster, a receiving fraudster and intermediaries, i.e. money mules or smurfs, involved in an* ML activity. *At that, the* length *of a fraud chain is determined by the number of money mules performing fraudulent transactions.*

In ML, fraudsters can act in organized groups to perform more complex financial transactions providing a better camouflage to the illegal source, i.e., a fraud chain can involve several sending and receiving fraudsters. In order to reflect the structure of the organized group performing ML activities, we use a fraud chain configuration.

**Definition 2 (Fraud chain configuration)** *A* configuration *of a fraud chain is determined by the number of sending and receiving fraudsters participating in this fraud chain. Four following configurations are possible:*

1. one-to-one – *the chain has one sending and one receiving fraudster;*

2. one-to-many – *the chain has one sending and multiple receiving fraudsters;*
3. many-to-one – *the chain has multiple sending fraudsters and one receiving fraudster;*
4. many-to-many – *the chain has multiple sending and receiving fraudsters.*

*Based on the configuration, we distinguish between simple fraud chains and combined fraud chains. A simple fraud chain has the configuration one-to-one. A combined fraud chain can have configurations 2, 3, or 4 and consists of two or more branches represented by simple fraud chains. The branches can share money mules, or use different intermediaries.*

**Definition 3 (Money Laundering activity)** *A Money Laundering (ML) activity of a fraud chain can consist of one or more ML operations occurring at arbitrary time intervals during the observation period.*

**Definition 4 (Money Laundering operation)** *Each Money Laundering (ML) operation represents a complete structured money transfer between two fraudsters and consists of several individual ML transactions between the fraudsters and the mules belonging to the fraud chain.*

If fraudsters want to act smart and use different mules for every ML operation, then, the longer the observation period, the higher the length of the fraud chain is.

We consider an ML scenario as depicted in Figure 4.2. Fraudsters Mallory and Oscar are end-users of an MMT service.

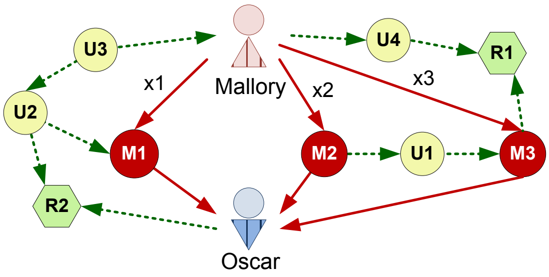


Figure 4.2: Fraud Scenario “Smurfing”

Mallory needs to transfer a large amount $X$ of mMoney to Oscar and does not want to leave any direct traces between their mobile accounts that can be logged by the service platform or trigger an alarm on exceeding transaction amounts. Mallory recruits, e.g. using phishing, $n$ end-users as money mules, who form her network of intermediaries participating in ML. For each ML operation, Mallory selects an arbitrary number of mules from this network and transfers to each mule $M_i$ a small share $x_i$ of the total mMoney amount to be “laundered”, such that $\sum x_i = X$. In Figure 4.2, the six corresponding ML transactions are denoted with solid (red) lines. Mules keep a small fixed percentage ($\leq 10\%$) of the received mMoney as a service fee and transfer the rest to Oscar.

The explored fraud scenario involves following assumptions: (i) fraudsters and mules can make regular transactions (shown with dotted (green) lines in Figure 4.2), such as MD or C2C transfers, along with ML transactions; (ii) mules do not make fraudulent transfers among themselves.

## 4.2. Background

### 4.2.1. Fraud Detection Data

Fraud detection data can either be real or synthetic ones. Most studies on fraud detection [150, 156] are based on proprietary databases and thus cannot be used for comparison of fraud detection techniques. The responsible parties may be reluctant to disclose information about vulnerabilities of their services and are obliged to maintain privacy of their clients. In order to create public databases for fraud detection, synthetic data were produced [157, 158, 159]. Simulators from [157, 158] are dedicated to fraud in video-on-demand systems and are not applicable in our case. The synthetic database [159] targets MMT systems. However, it does not model the MMT architecture and ML techniques such as smurfing. The simulator introduced in [160, 65] enables the simulation of more complex scenarios such as ML activities. For this reason, we adapt for the MMT simulator used to create synthetic databases for ML detection (cf. Section 4.4.1).

### 4.2.2. Fraud Detection Techniques

Many techniques were investigated for fraud detection, mainly, from the statistical and data mining field [150, 151, 152, 153]. The easiest way is to use thresholds on transaction amounts or any other statistical value, such as transaction or expenditure rate [150]. Among data-mining techniques, mostly used are neural networks, SVMs, Bayesian network models, and naive Bayes scoring. We can also mention decision trees, decision tables and logistic regression, which are easier to interpret than neural networks or SVMs. A comparative study of SVM, random forest and logistic regression on a database of real-life credit card transactions in [153] revealed that while SVM and logistic regression showed good results, the random forest technique had the overall better performance. A comparative study of several machine learning algorithms on a synthetic database of MMT transactions in [160] presented the fraud cases related to theft and malicious infections of mobile phones. The studied algorithms are Bayes net, naive Bayes, SVM, regressions, nearest neighbors, decision table, decision tree and random forest. The best performing algorithms in this study were PART decision table, C4.5 decision tree and random forest. The work [159] showed that random forest provided better results than naive Bayes classifier and random tree on a synthetic database of MMT transactions containing ML. The survey [152] covered such methods as SVM, decision trees (CART, C4.5, C5.0), neural networks, Bayesian belief networks, hidden Markov models and link analysis as well as unsupervised techniques, such as anomaly detection and clustering. Authors in [150] reviewed fraud detection methods, such as rule-based methods or link analysis. However, these works did not provide a clear comparison or recommendations regarding these techniques.

### 4.2.3. Fraud Detection using Process Analysis

Formal methods, such as linear temporal logic, state-charts, and related formalisms have been applied for runtime monitoring of concurrent distributed systems in [161, 162]. However, these works are mainly aiming at error detection, for example, concurrency-related bugs. A classification for runtime monitoring of software faults is given in [163]. In [164], the use of event-triggered rules for sensing and responding to business situations is introduced. A formalized approach to security risk modeling for electronic business processes in [165] comprises simulation aspects, but not the usage of runtime models. Process mining techniques for analysis of large data sets and data streams aim to extract valuable process information building on techniques from data mining and machine learning, where knowledge discovery from data is mostly based on various statistical methods [166]. According to a classification of approaches in the field of business process management in [167], our work implements the "check conformance using event data" approach, where information from the process model and the event data is used to identify deviations of runtime behavior from expected behavior. The work on runtime compliance verification for business processes in [168] is complementary to this approach.

To the best of our knowledge, the only application of business process analysis to fraud detection in MMT systems was reported in our previous work [17]. A process model reflected transfer habits of end-users. The detection was based on the assumption that for each end-user the transaction amount is limited to a constant range and does not suddenly change. Amount classes were defined and transitions between these classes were monitored. If an abnormal change was observed, an alert was generated and the transaction was labeled as fraudulent. The fraud scenario implied that fraudulent transactions have much lower amounts than the average in the system. The work showed a good performance with the recall of 80%-90% on modeled transactions and 40%-45% on all fraudulent transactions.

## 4.3. Model-based Fraud Detection in MMT Service

### 4.3.1. Design Goals

Common techniques for fraud detection in the banking field show substantial limitations when applied to MMT. Since in most cases supervised methods are used, a good training database is vital for the reliable performance. The problem is that for MMT services these data are usually not available. The chosen fraud scenario poses additional difficulties: fraudsters tend to camouflage ML activities to make them statistically indistinguishable from the behavior of a regular user. We aim to provide an alternative method for fraud detection in MMT services achieving the following:

**Recognition performance.** The method is intended to support analyst activity and should offer recognition performance comparable to the state of the art. Considering that parameters of ML operations may be very close to those of legitimate transactions *false positives* are held acceptable and deserving further investigation. *False negatives*, on the contrary, are critical, because fraud committed with an MMT service can have legal implications for its provider.

**Usability.** The method should increase analyst efficiency. Usability in this case means that the method should be easy to use, give meaningful alerts and reduce total alert volume, offer comparable or better performance than traditional fraud detection techniques. Alert reduction is an important goal because the extensive number of alerts produced by a fraud detection system affects reaction times and hinders its adoption. For the same reason, alerts need to be easily interpretable and signify actionable results. At that, detection delay – the time span between a fraudulent event and its recognition – is a major performance metric and should be minimal.

**Autonomy.** The method should not depend on availability and relevance of training data and signature bases. For ML, real samples of fraudulent operations are often unavailable. Though simulated datasets can be used in some cases (cf. Section 4.2), we believe that such dependency would limit the adoption. Signature-based methods use preset fields that must be met to trigger a rule. As ML schemes are diverse and flexible, the latter is also considered restrictive.

Next we introduce our method for detection of fraud chains related to ML; its pros and cons in comparison with classical machine learning techniques will be discussed in Section 4.4.4.

### 4.3.2. Predictive Security Analysis at Runtime

As a basis for our fraud chain detection method we adopted a model-based approach for event-driven process security analysis PSA@R [15]. Here we summarize the concepts used for the analysis of the MMT system.

The core idea of PSA@R is to validate security compliance of critical processes, by evaluating events related to their execution against formally defined workflows and security properties of these processes. It enables identification and management of changes in process behavior as well as early detection of possible security requirement violations for proactive response. Figure 4.3 shows three major phases of PSA@R in the application to an MMT system. Firstly, at the *specification* phase, chosen MMT processes need to be formally defined. At that, three interrelated formal models are created: process model, event model and security model. An operational *process model* is specified using Asynchronous Product Automata (APA), a family of elementary automata [169]. PSA@R uses an operational formal model of a process to compute its expected behavior depending on the observed system state. The process behavior is represented as a directed Reachability Graph (RG) of an APA, whose nodes refer to states and labeled edges to state transitions of the APA. State transitions are driven by (internal) events extracted from the input event stream. An *event* implies that a certain process *action* has been executed resulting in a new state. An *event model* determines the internal mapping for the runtime events defined by an event schema. A *projection* associated with the event model maps real events (MMT transactions) to abstract internal events by filtering out information that is not relevant for security analysis. A *security model* specifies process security properties by means of finite-state automata, so-called monitor automata [15]. Monitor automata specify the requirements by means of predicates annotated at the edges of a monitor automaton, which express the required properties in terms of state transitions in the current or predicted behavior (given by the RG). Security critical states of a monitor automaton signify violations of security requirements.

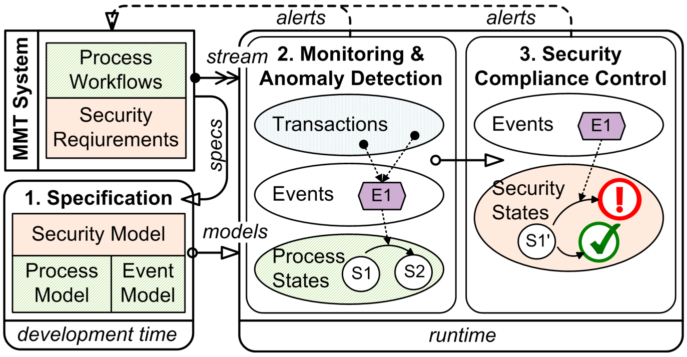


Figure 4.3: Phases of Predictive Security Analysis at Runtime in MMT System

At runtime, PSA@R performs *monitoring and anomaly detection* in the MMT system. In order to verify the actual process behavior, events from the input stream representing actions of the MMT system are checked against the process model of the originating process instance. PSA@R identifies deviations from the expected workflow and produces alerts. For *security compliance control*, PSA@R validates if the actual process behavior meets the specified security properties. If an event triggers a state transition in one of monitor automata representing security properties, the state of the automaton changes accordingly. In case a critical state is reached, a security alert is generated. If PSA@R finds within the prediction scope a possible state transition of a monitor automaton which leads to a critical state it generates a predictive alert (warning). Figure 4.4 illustrates the key notions of the method described above.

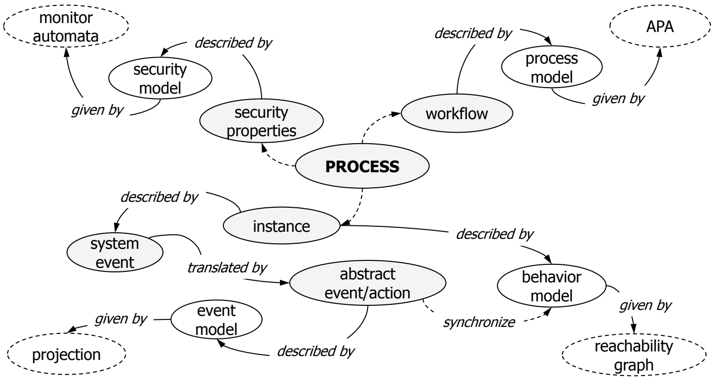


Figure 4.4: Main Notions of Process Modeling

PSA@R requires as input only descriptions of process workflows that need to be verified at runtime and corresponding security requirements, and does not use any other prior information, such as samples of ML transactions. The tool implementing this method traces all references during runtime and thus produces easy interpretable alerts, since they are linked to the states of the respective process and security models. For the purpose of ML detection, we extend PSA@R with an additional method allowing to reveal fraud chains. We describe our contribution in the next section.

### 4.3.3. Revealing Fraud Chains

Detection of fraud chains in MMT services is a new application scenario for PSA@R. One of the main challenges in adoption of a model-based approach is to find a proper abstraction level to define processes regarding ML activity, otherwise the performance and usability are affected [17]. For this reason, we had to extend PSA@R, namely, to add a capability to define and synchronize processes on different abstraction levels using synthetic events derived from the monitoring data. Following PSA@R, we describe underlying formal models and then introduce an algorithm which allows us to identify fraudsters among end-users and reconstruct chains of money mules used for ML. Further we refer to the proposed detection method as the FCD (Fraud Chain Detection).

**Event Model.** We focus on the suspicious behavior of end-users observed from incoming and outgoing transactions on their mobile accounts. Events are propagated when an end-user commits a transaction. The transactions log defines the format of events received from the MMT system and contains for every transaction $T$ the sender $s$, receiver $r$ and amount $a$, along with other fields $p_i$, omitted in the current study. We define the event mapping as follows: $T(s, r, a, p_i) \rightarrow \mathcal{E}(s, r, a)$. Every internal event $\mathcal{E}(s, r, a)$ generates two actions: the $send(\mathcal{E})$ action related to the sender of the transaction $T$ and the corresponding $receive(\mathcal{E})$ action for the receiver of this transaction.

**Process Model.** We define an abstract ML process that represents ML activity of a fraud chain. Each state of this process refers to a money transfer between two fraudsters made through the mediation of a mule. Figure 4.5b shows an RG for the ML process: the more nodes the graph has, the more intermediary-enabled transfers were performed. Thus, the number of the nodes determines the length of a fraud chain. State transitions in this process model are driven by events representing transactions committed in the MMT system. The edges of the RG are labeled with the respective actions $send$, $receive$, and $laundering$ (cf. Figure 4.5b). The actions $send$ and $receive$ correspond to an *individual* process that represents the behavior of end-users conducting c2c transfers. Instances of this process are characterized by the user identifier available as an attribute of the event, i.e. $s$ or $r$. At that, an observed MMT event changes the current process state for both the sender and the receiver. Monitoring of individual processes allows us to single out mule candidates and potential fraudsters, as senders and receivers of transfers made with mWallets of the supposed mules. A *network* process represents the behavior shown by a group of end-users, in our case, by a pair of fraudsters, who organized a fraud chain. An identifier of a network process instance is an identifier of the fraudster pair. Multiple instances of the network process refer to the same state of the abstract ML process. When a new mule candidate appears, the synthetic $laundering$ action is generated, and the respective network process proceeds to a new ML state (see algorithm FCD).

**Security Model.** We set a security goal for the individual processes as follows: *Mobile accounts in an MMT service owned by end-users must not be used to conduct money transfers on behalf of a third party.* To determine if the observed behavior of an end-user can be rated as ML activity and single out mule candidates we use the following criterion:

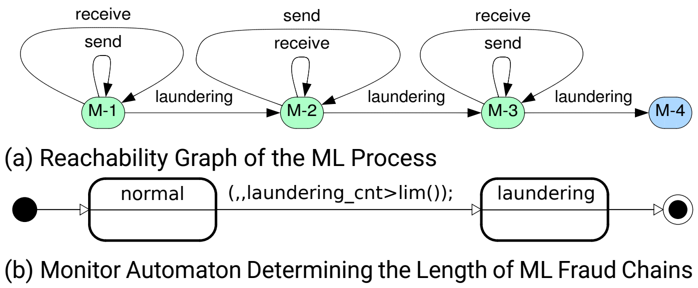
**Criterion 1** *If for an mWallet of an end-user an outgoing c2c transaction $send$ with the amount $a_{sent}$ and a previously committed incoming c2c transaction $receive$ with the amount $a_{rec}$ : $a_{rec} - a_{sent} \leq \Delta_a$ exists, then the user owning this mWallet is labeled as a mule candidate; the sender $s$ of $receive$ and the receiver $r$ of $send$ are labeled as fraudster candidates.*

The parameter $\Delta_a$ is a service fee charged by mules. It can vary between fraud chains, but is constant for all ML operations performed by the same chain. In accordance with [170], we limit the fee to $0 < \Delta_a \leq 10\%$ in our experiments (cf. Section 4.4.2).

For the network processes, we formulate the following security goal: *End-users of an MMT service must not conduct structured money transfers involving intermediate mobile accounts.* This goal is intended to rule out the behavior that helps to disguise the original source and total amount of mMoney transfer. To decide if end-users suspected to be money mules are engaged into the same fraud chain we evaluate the criterion:

**Criterion 2** *$f1$ and $f2$ are fraudster candidates. If for each of the two mule candidates $m1$ and $m2$ an outgoing c2c transaction $send : r = f2$, and a previously committed incoming c2c transaction $receive : s = f1$ exists, and $\Delta_a^{m1} = \Delta_a^{m2}$, then both mule candidates $m1$ and $m2$ belong to the fraud chain $(f1, f2)$.*

Generally speaking, it is possible that in an MMT service legitimate chains emerge. For example, parents can transfer money to their child, so that s/he is able to pay the rent to the landlord. But the length of such chains will be usually short (in this scenario it equals 1). For this reason, we introduce a *detection threshold* to enable the control over generated alerts. The detection threshold defines the number of intermediaries involved into transfers between two end-users. The higher this number, the more likely the observed activity is ML. The monitor automaton representing this condition is given by Figure 4.5a, where $lim()$ refers to the detection threshold.



(a) Reachability Graph of the ML Process

(b) Monitor Automaton Determining the Length of ML Fraud Chains

Figure 4.5: Process Model and Security Model for ML Detection

**Algorithm FCD.** An RG that describes the common behavior of MMT end-users as well as the behavior of the parties involved into ML is computed (see Figure 4.5b). During simulation, multiple process instances reflecting the end-user behaviors and ML activities are assigned to this graph. The detection algorithm verifies whether transitions in this processes occur. As defined in the event model, every transaction (MMT event) is mapped to the internal event $\mathcal{E}(s, r, a)$ that generates the $send(\mathcal{E})$ action and the corresponding $receive(\mathcal{E})$ action. For every $receive(\mathcal{E})$ action the respective transactions are stored in the transactions table $RT$ under the identifier of the receiver $r$ of the event $\mathcal{E}$: $\mathcal{RT}[r] := \mathcal{RT}[r] \cup \mathcal{E}$

Based on the data stored in the element $\mathcal{RT}[s]$ of the table $\mathcal{RT}$ for every $send(\mathcal{E})$ action ML candidates are determined by means of the function $getlc$. The result of this function is a list of sending fraudster candidates. In the function $getlc$ a heuristics for the candidate selection is implemented based on Criterion 1. In the laundering table $\mathcal{LT}$ all transactions related to the particular fraudster pair $(f1, f2)$ are stored. The function $check\_laundering$ implements a heuristics defining whether the laundering process for a fraudster pair continues, as given by Criterion 2. The helper functions $getr$ and $gets$ deliver the identifier of the receiver and the sender for a transaction respectively. The pseudo code 1 describes the processing of the $send(\mathcal{E})$ action on an abstract level.

```
for $f1 \leftarrow getlc(\mathcal{E}, \mathcal{RT}[gets(\mathcal{E})])$ do
    $f2 := getr(\mathcal{E})$
    $lid := (f1, f2)$
    if $check\_laundering(LT[lid], \mathcal{E})$ then
        $LT[lid] := LT[lid] \cup \mathcal{E}$
        $generate\_action(laundering, lid, \mathcal{E})$
    end
end
$generate\_action(receive(\mathcal{E}), getr(\mathcal{E}), \mathcal{E})$
```

**Algorithm 1:** FCD

The function $generate\_action$ generates a new action (1st parameter) for a given process (2nd parameter) and is responsible for state transitions in the RG for this process.

## 4.4. Experiments

### 4.4.1. MMT Simulations

We use the MMT simulator from [65, 160] to generate a database needed to conduct experiments on ML detection. As the simulator presented in [159], this one is based on a multi-agent platform. However, it simulates both the MMT system and users of this system, as well as the habits of end-users. The simulated platform is made of (1) a front office which interacts with users and processes operation requests and connections to the service, (2) an account management system which controls accounts and processes financial operations, (3) a logs server and (4) a data warehouse which registers the history of the front office and the account management. The payment sequence expects the following pattern [171]: (1) authentication, (2) transmission of sender's payment instructions and transaction details to the MMT platform, (3) authorization by the MMT platform, (4) credit and debit on the receiver's and sender's accounts. A log entry is created when a simulated user carries out a transaction and registered in the transaction database, after the account management system. Each entry contains the transaction type, the transaction amount, the sender and receiver pre- and post-transaction balance, the sender and receiver category, and the transaction date. The generated database contains all simulated transactions for several months.

Three categories of legitimate actors are involved in the MMT system: *End-users*, *Merchants* and *Retailers*. Each category consists of several roles that are associated with specific actions in the platform. *End-users* are individuals who use their mobile devices to access the MMT platform and carry out transactions. *Merchants* sell services or goods to end-users. *Retailers* are in charge of the distribution of electronic money. The simulation is based on the assumption that legitimate users' transactions are mostly related to their habits. A habit is a repetition of a sequence of legitimate transactions which are characterized by (1) a type of transaction, (2) a normally distributed transaction amount, (3) a normally distributed period of time between two transactions of the considered habit, (4) an initial date and (5) a final date. A user's behavior is composed of a set of habits $H = \{H_1, ..., H_i, ..., H_n\}$, where $H_i$ is a habit for one specific type of transaction. Habits assigned to end-users come from a list of five available habits: Money Deposit (MD), Money Withdrawal (MW), Merchant Payment (MP), Client-to-Client transfer (C2C) and Airtime Recharge (AR).

The malicious behavior of fraudsters and mules is also modeled as habits. Thus, fraudsters and mules are selected randomly from the end-users, and the respective habit is added to their habits. Each sending fraudster is associated with a list of mules representing the mules recruited by the fraudster. The behavior of the sending fraudster is to launch ML operations on a regular time basis. To launch an ML operation, she chooses several mules from the list, splits the amount of money to be laundered, and sends the money to the chosen mules within a short interval of time. On receiving the money, a mule transfers it to the receiving fraudster within a day keeping a fee.

**Configuration of the MMT Simulator** We created 10 000 end-users, who have between 1 and 4 habits. Table 4.1 presents the percentage of end-users with respect to the number of habits they have. The table also shows, which habits are associated with end-users depending on the number of their habits. For example, the majority 63,17% of created end-users have only 1 habit and 26,30% have 2 habits. The AR habit is shown by 60,35% of the users with 2 habits. According to these figures, end-users with 1 habit mostly conduct AR and few MD, while those with two or more habits use much more MD and C2C.

Table 4.1: Partition of End-users and Habits by the Number of Habits

| | 1 habit | 2 habits | 3 habits | 4 habits |
|---|---|---|---|---|
| Part of end-users | 63,17% | 26,30% | 8,67% | 1,86% |
| MD | 11,54% | 82,37% | 97,66% | 98,91% |
| MW | 2,76% | 18,86% | 46,73% | 97,83% |
| MP | 0,22% | 2,46% | 3,50% | 6,52% |
| C2C | 2,79% | 35,95% | 63,55% | 100% |
| AR | 82,69% | 60,35% | 88,55% | 96,74% |

From among these end-users, we created 10 fraud chains made of a sending fraudster, a receiving fraudster, and several mules. All these parties are chosen randomly among the end-users. Each fraud chain has a different number of mules, and a varying number of mules is used for ML operations. This configuration is presented in Table 4.2. For example, fraud chain 7 has 7 mules, but only 4 of them, randomly chosen, are used for each ML operation. The fee rate is fixed randomly, but is the same for all mules from the same fraud chain. The sending fraudster conducts an ML operation approximately each month.

Table 4.2: Partition of Mules Number among Fraud Chains

| Fraud chain | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| No. mules recruited | 3 | 3 | 5 | 5 | 5 | 7 | 7 | 7 | 7 | 7 |
| No. mules used | 3 | 3 | 3 | 4 | 5 | 3 | 4 | 5 | 6 | 7 |

Two databases were created. The first, database A, is used for the training phase of machine learning algorithms. It covers four months of data and contains 272 038 transactions, among which 329 are fraudulent representing 38 ML operations. The second, database B, used for the validation of ML detection techniques, represents seven months and contains 466 359 transactions, with 611 fraudulent ones for 72 ML operations. Figure 4.7a shows when each fraud chain performed an ML operation. Each cross on the figure represents an ML transaction and each cluster of crosses represents an ML operation. We can see that there is approximately a month between the ML operations. The ML activity of each fraud chain lasts throughout the seven months of the simulation.

### 4.4.2. Experiment Setup

The goal of our experiment is to compare the efficiency of several machine learning algorithms and the new FCD method for ML detection, using the same database generated by the simulator. To evaluate the applicability of our model-based method for fraud chain detection, we implemented it as an AML plug-in to the process modeling and verification tool PSA [34], [17]. For the machine learning algorithms, we used the Weka toolbox [172]. We chose the PART decision table [173], the C4.5 decision tree [174] and the random forest algorithm [175] (cf. Section 4.2).

The machine learning algorithms used a data format aggregated from the original transaction logs. We added fields computed over time to have more information on each transaction. We computed the minimum, maximum, mean, and the total amount of transactions emitted by the sender within a week, as well as the number of transactions, and the number of transactions with the receiver. We did the same data aggregation over a day and an hour. We also kept some original fields, such as the type and amount of the transaction, the sender's and receiver's category, and the date of the transaction. This format was proved to be more efficient for machine learning algorithms than the original one [160].

The machine learning algorithms are trained on database A and tested on database B. The FCD is tested on database B. The performance metrics for the evaluation are precision and recall regarding fraudulent transactions. These metrics are extracted from the confusion matrix presented in Table 4.3. The precision $\frac{TP}{TP+FP}$ should be possibly high to avoid false alarms, which require time to investigate and might block legitimate end-users in real MMT systems. The recall $\frac{TP}{TP+FN}$ should also be high, since it means that a fraud chain is discovered faster.

Table 4.3: Confusion Matrix

| | Predicted normal | Predicted fraudulent |
|---|---|---|
| Actual normal | True negative (TN) | False positive(FP) |
| Actual fraudulent | False negative (FN) | True positive (TP) |

### 4.4.3. Results

We present results of the three selected machine learning algorithms in Table 4.4. The figures reveal that the C4.5 decision tree and the random forest work better than the PART decision table. The C4.5 decision tree and the random forest show roughly the same results, with the precision around 97% and the recall around 37%. However, the C4.5 decision tree has slightly better performance. Even though the precision is good, the recall is quite low for all three algorithms. A closer look at this result suggests that for all ML operations at least one transaction is labeled as fraudulent. Figure 4.7b depicts the result of the PART decision table algorithm. Each cross corresponds to a true positive, and circle to a false positive. False negatives are not presented for readability reasons.

Table 4.4: ML Detection Results for Machine Learning Algorithms

| | PART | | C4.5 | | Random Forest | |
|---|---|---|---|---|---|---|
| | N | F | N | F | N | F |
| Actual normal | 465 721 | 27 | 465 741 | 7 | 465 740 | 8 |
| Actual fraud | 397 | 214 | 381 | 230 | 385 | 226 |
| Precision | 88.79% | | 97.04% | | 96.58% | |
| Recall | 35.02% | | 37.64% | | 36.98% | |

Comparison of Figure 4.7a and Figure 4.7b shows that all 72 ML operations have been detected, but not all transactions from an ML operation were labeled. In general, the first two transactions from the sending fraudster to mules are not classified as fraud, neither the transactions from mules to the receiving fraudster. This explains the recall of 35%. Thus, it would require additional efforts to detect the complete fraud chains. Results for C4.5 detection tree algorithm are presented in Figure 4.7c. Similarly, each ML operation can be detected with some investigation. Results of the random forest,not pictured here, are very close to those of the C4.5 algorithm.

Table 4.5: ML Detection Results for the FCD

| | Normal | Fraudulent |
|---|---|---|
| Actual normal | 465 747 | 1 |
| Actual fraudulent | 60 | 551 |
| Precision | 99.81% | |
| Recall | 90.18% | |

Table 4.5 shows results for the FCD acquired with the detection threshold of 3. The precision and recall are, respectively, of 99.8% and 90.1%, which is a way better than with machine learning algorithms. In Figure 4.7d, showing the result for PSA@R, the crosses denote positively labeled transactions. Thus, the cross on the FP line is a false positive, while the others are true positives. The circle corresponds to the negative detection, so the circles on each fraud chain line are false negatives. We can see that with the detection threshold of 3 the FCD can miss ML operations (chains 1, 2, 3 and 6) if the fraud chain length is lower than this threshold ($< 3$). Notably, once a fraud chain is detected, all subsequent transactions are correctly detected as fraudulent. In Figure 4.7d, a time interval before fraud chain is detected and fraudulent transactions can be identified is denoted as $t_{psa}$. This interval depends on the chosen detection threshold and can be shortened by lowering its value. We suppose that in this case there might be a trade-off between detection delay and false positive rates, but we could not prove this consideration on our testing data.

We also investigated the effect of sliding time windows on the FCD detection capabilities. As the number of end-users in an MMT service is usually high, we addressed a scenario, when the proposed method meets its performance limits. One of possible solutions in such situation is to restrict the number of monitored ML processes to a sliding window, and discard candidates that are outside this window (cf. Section 4.3.3). We made tests with sliding windows of sizes from one to six months (cf. Figure 4.6). In all cases, the FCD was able to correctly detect fraud chains. Thereby, the smaller

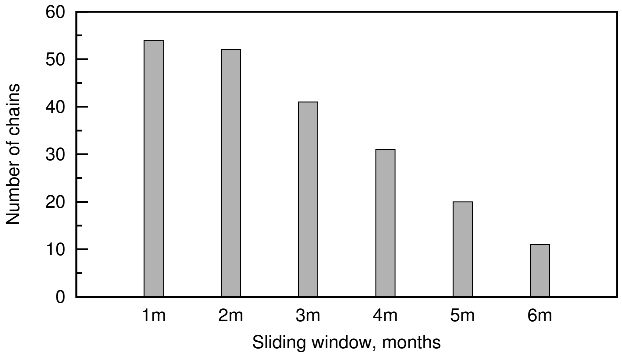


Figure 4.6: Effect of Sliding Time Window on the FCD Fraud Chain Detection

the size of the sliding window, the more fraud chains are detected. The reason is that for the same pair of fraudsters all mules involved in transactions over a longer observation period are ascribed to the same chain, while for a smaller sliding window a new chain is created. Thus, the number of detected chains is larger, while the chains themselves are shorter.

### 4.4.4. Discussion

Our analysis proved both machine learning algorithms and the new FCD method to enable the detection of fraud chains related to ML activity. The precision is good in both cases, and only few false positives appear, even though simulated ML operations have parameters very close to regular transfers. However, we see several advantages of our chain detection method based on PSA@R. First, according to the overview of existing fraud detection techniques in Section 4.2, semi-supervised or unsupervised approaches should be preferred for operational reasons. This holds for FCD, whereas supervised machine learning algorithms rely on a training database to work. In this sense, the proposed method is autonomous, depending only on specifications of processes, which security compliance needs to be verified. The FCD also demonstrates better precision and recall than machine learning algorithms. Thereby, machine learning algorithms can only label individual transactions leaving the task of fraud chain detection to an analyst, while the FCD singles out fraud chains immediately. This leads to much less effort during the investigation of alarms raised by a fraud detection tool. Such system has a better usability and is more economic for MMT service providers.

We see a potential advantage of machine learning algorithms in that suspicious activity can be detected earlier, although with an additional investigation overhead. The comparison of ML detection results for both approaches presented in Figure 4.7 shows that with the FCD there is a delay between the beginning of an ML activity and its recognition. During this time span the respective fraud detection tool would not give any hints about the conducted fraud. This delay could be minimized by selecting a proper detection threshold. In this respect, we analyzed the influence of the fraud chain length. We found that with few mules in the chain, it might be more complicated to detect the chain, and a lower detection threshold should be set. However, the higher the amount of money a fraudster wants to transfer, the larger the chain length becomes. Thus, a critical ML activity can be detected already on the first ML operation.

The computational performance of both approaches is satisfying, with analysis of all 466 359 transactions (10 000 end-users) of database B done within minutes on a standard computer. However, considering that modern MMT services such as M-PESA number millions of users, we experimented with time sliding windows as a means to solve performance issues, which can occur when the FCD is deployed in a real MMT environment. The experiment proved that using a sliding window to reduce the number of simultaneously monitored processes can offer a plausible solution, since it does not affect the recognition performance.

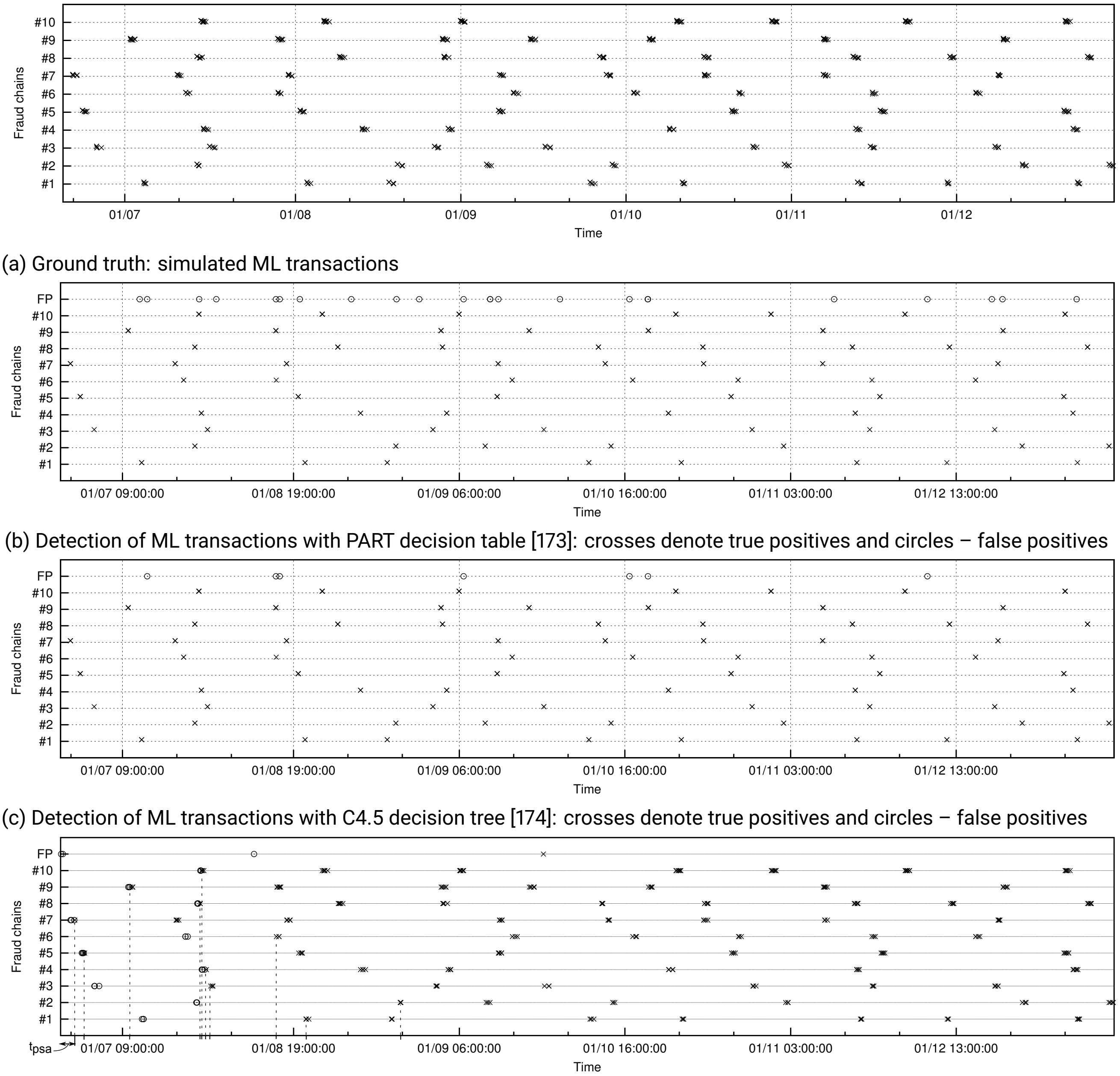
As no public data on ML were available, we evaluated machine learning algorithms and the FCD method on simulated transaction logs. Notably, using a simulated database can introduce a detection bias. Indeed, even though we reproduce the normal behavior correctly according to a real-world database, the fraudulent behavior is defined using a modeled fraud scenario. Such a bias can help detect the fraudulent cases. However, as all methods in the comparative study can exploit this bias, the comparison between algorithms is reasonable. An experiment with actual real-world data might present different results, but the order of algorithms regarding their performance should be the same.

We designed the FCD method bearing in mind the MMT use case. What makes ML techniques such as smurfing possible in MMT systems is the availability and ease-of-use of the MD, C2C and MW operations. These operations are not always available, e.g., the C2C in traditional e-banking. However, in case such operations are available, for example, in bitcoin transactions or V2V energy transfer, the proposed detection method is also applicable, provided it is tuned for the use case. Indeed, as users of a different service may behave differently compared to the MMT service subscribers, the transactions database might present different characteristics.

## 4.5. Chapter Summary

In this chapter, we describe a new model-based ML detection method that is able not only to identify individual fraudulent transactions, but detect complete fraud chains, i.e., end-users of an MMT service acting as fraudsters and money mules (or smurfs). This method extends the approach for event-driven process security analysis PSA@R [15] and enables its application to detection of fraud chains in ML scenarios. In order to prove the efficiency of our method we compared it with several classical machine learning algorithms using a synthetic database produced with an MMT system simulator. The recognition performance shown by the FCD method is better compared both to machine learning algorithms and business process analysis from [17], with precision and recall of 99.8% and 90.1% correspondingly.

Though with the fraud chain detection deployed, an MNO would be able to identify and potentially block phone numbers used by fraudsters, the prosecution would be possible only if the owner of this number can be found, which is often not the case [146]. For the future work, we plan to further extend this approach to detect other types of fraud conducted in MMT services, for example, agent frauds [176], and in other distributed payment services using digital currencies. We also plan to look into questions related to other fraud scenarios specific to Vehicle-to-Vehicle, Vehicle-to-Home, and Vehicle-to-Grid transactions. This includes simulation of the respective data sets and adaptation of the process models.



(a) Ground truth: simulated ML transactions

(b) Detection of ML transactions with PART decision table [173]: crosses denote true positives and circles – false positives

(c) Detection of ML transactions with C4.5 decision tree [174]: crosses denote true positives and circles – false positives

(d) Fraud chain detection with the FCD method: crosses denote positively labeled transactions, while circles – negatively labeled; $t_{psa}$ – detection delay

Figure 4.7: Comparison of ML Detection Results for Different Detection Approaches

# Part III.

# Platform Assurance in Safety-Critical Infrastructures

# 5. Enabling Trust in Vehicle-to-Grid Communications

In this chapter, we investigate the security of Electric Vehicles (EVs) identities and trust establishment with regard to Vehicle-to-Grid (V2G) applications. This chapter is based on on the previously published papers "TrustEV: Trustworthy Electric Vehicle Charging and Billing" [1], "HIP: HSM-based Identities for Plug-and-Charge" [7], and "HIP-20: Integration of Vehicle-HSM-Generated Credentials into Plug-and-Charge Infrastructure" [8].

Electric mobility (or e-mobility) experiences a comeback as one of the major transportation technologies to cut CO2 emissions for climate protection, to improve air and noise pollution in cities, and to enable further advancements such as autonomous vehicle operation. E-mobility encompasses all vehicles used on public roads that are propelled by an electric motor and equipped with a rechargeable battery as their main energy resource, i.e., EVs. The adoption of EVs beside their relatively high price and a narrower selection of models compared to conventional cars is slowed down by a complicated and time-consuming charging process [118]. Since private charging is seldom available, the public charging plays an important role for e-mobility. In order to enable EV charging in public environments, a complex infrastructure has emerged that involves energy providers, Distribution System Operators (DSOs), E-Mobility Service Providers (eMSPs), Charge Point Operators (CPOs), as well as roaming and billing service providers. To be able to recharge their EVs at a public Charge Point, drivers in most cases need a service contract with an eMSP and have to authenticate themselves each time before the charging starts using an RFID card or a smartphone app provided by this particular eMSP. It is common for drivers to carry with them a collection of RFID cards and apps to use charging services at various locations. Ad-hoc charging with a credit or debit card is expensive for providers [70] and, thus, not usually available at the moment[1]. These methods are not user-friendly and also are not compatible with newer use cases like automated parking or autonomous driving, because they require user interactions.

Beside providing charging services to the drivers, EV charging has another critical aspect. During charging, EVs are connected to the local electric grid as an additional unaccounted load that can considerably increase (peak) power demand through an uncoordinated consumption and potentially lead to the grid overload. In order to mitigate these negative effects and to provide grid support required for load management during peak times or power supply failures, the V2G technology was developed. V2G allows an EV to communicate with the power grid to optimize its charging behavior depending on the current demand, e.g., to limit the charging rate or even discharge the battery and deliver electricity to the grid when demand is high.

In order to address all these aspects, the international standard ISO 15118 "Road vehicles – Vehicle to grid communication interface" [37, 38] was developed that defines the V2G communication interface for the bidirectional energy transfer between the EV and Charge Points (or Electric Vehicle Supply Equipment (EVSE)). This standard allows the Electric Vehicle Communication Controller (EVCC) and the Supply Equipment Communication Controller (SECC) to exchange information on the vehicle's state-of-charge, planned departure time, location and further charging-related parameters that can be used by network operators to calculate a charging profile for this vehicle in such a way that optimally balances the local grid. Beside enabling the V2G functionality, ISO 15118 aims to largely automate the actions necessary on the side of drivers to charge their EVs by introducing the Plug-and-Charge (PnC) mechanism. In particular, the standard automates the authentication of EVs, the authorization of the charging session, and the collection of billing information. When using the PnC, the vehicle only needs to be connected to a power socket, by hand or with the help of a charging robot [177], or parked over a coil for inductive charging, with the rest being handled by the vehicle itself transparently for the driver.

Since ISO 15118 describes only the communication interface between the vehicle and the charger, further communication protocols to connect the charger and the backend systems of service providers were developed (cf. EV Related Protocol Study [40]). For example, all data necessary for authorization and billing of charging sessions (e.g., energy meter values) are transferred between the Charge Point and the PnC backend systems via Open Charge Point Protocol (OCPP) [80, 83, 130, 85]. Notably, an eMSP usually cannot connect directly to EVs of its customers or to Charge Points and relies on intermediate parties during credential provisioning and billing.

[1] In Germany, a new regulation mandates this method for all new Charge Points installed after 1.7.2023 (cf. `https://www.bmwi.de/Redaktion/DE/Pressemitteilungen/2021/05/20210512-spontanes-laden-von-e-autos-wird-einfacher.html`)

The ISO 15118 standard employs public-key authentication based on X.509v3 certificates and the dedicated V2G Public Key Infrastructure (PKI) (cf. [38], Annex E). The mechanism for credential provisioning specified by the standard includes two major phases. During production each ISO 15118 enabled EV receives a lifelong identity for PnC from its Original Equipment Manufacturer (OEM), so-called OEM provisioning credential, which the driver registers at an eMSP when concluding a charging service contract. On first charging, the EV uses this identity to install the contract credential provided by the eMSP, which is the actual credential used for PnC authentication. The application guideline VDE-AR 2802 [78] for handling contract credentials focuses on backend aspects of credential provisioning and presents the out-of-band delivery of contract credentials.

These PnC credentials are an attractive target for adversaries and must be protected against illegitimate access and misuse. The respective keypairs are created by the OEM and the eMSP and are at least temporary stored in their backend systems before being installed into the vehicle or encrypted for the delivery. Thus, a successful attack on the respective corporate networks may allow an adversary to get access to these keys. Several examples of this type of attacks are known, such as the Sony hack from 2014 [53], or DigiNotar and Comodo breaches from 2011 [54, 55]. EVs and Charge Points are also vulnerable to malicious attacks. Both local and remote attacks on chargers from multiple manufacturers have been recently demonstrated [56, 57, 58]. Similarly, car hacking research [59, 60, 61, 62] shows that an adversary can get access to basically any Electronic Control Unit (ECU) in a vehicle. This becomes especially critical considering that ISO 15118's security concept only holds under an implicit assumption that proper credential management is in place and the keys are protected from disclosure both in the respective backend systems and the vehicle itself [50].

However, ISO 15118 currently demands only basic protection of the data transfer by using Transport Layer Security (TLS) and do not specify any system security requirements. The standard neither defines key handling in the OEM's or eMSP's backend, nor how keys are protected or under which circumstances credentials should be revoked. With ISO 15118 being actively adopted worldwide by OEMs and service providers in automotive and energy sectors [77], the security of PnC and V2G in general becomes even more critical for e-mobility and energy supply. Currently, a new edition ISO 15118-20[2] is under development and changes to the security concept can be proposed.

A standard solution to protect sensitive cryptographic keys and perform cryptographic operations is to use a Hardware Security Module (HSM). In the automotive field, several HSMs were specified. Examples for automotive HSMs include the Secure Hardware Extensions (SHE) module [178], the EVITA HSMs [179, 180] project, or HSMs for the V2X communication from Car 2 Car Consortium [181]. However, only EVITA HSM-full provides the cryptographic primitives required for PnC by the specification ISO 15118-2[3] [38] but exists only as proprietary solutions and has no specific security certification. For this reason, the Trusted Platform Module (TPM) 2.0 [182] specified as an open standard by the Trusted Computing Group (TCG) and approved as an international standard [115] gets more attention in the automotive context. For example, Volkswagen integrates dedicated hardware TPMs into their cars [132]. These TPMs with their shielded location are Common Criteria (CC) EAL4+ certified and qualified according to the automotive AEC-Q100 standard making them a good candidate to secure critical credentials. Beside standard HSM functionality of secure key storage and secure execution environment for cryptographic operations, the TPM 2.0 provides additional security mechanisms, including measured boot, enhanced authorization allowing for arbitrary security policy statements, and credential protection [113]. From the implementation perspective, the TPM's advantage is the availability of the hardware (even for general-purpose development platforms) and the availability of the open source TPM Software Stack (TSS)[4].

Since no publicly available concepts or implementations using an HSM or a TPM in this context exist, one of the goals of this work is to analyze the implications of using the TPM 2.0 to secure ISO 15118's PnC and whether it can be used in a standard conform manner. Although the TPM is an established technology with some automotive applications, its integration into the processes of ISO 15118 is not straightforward. For one thing, the ISO 15118 specification defines strict timing and data size constraints for the communication between the vehicle and the charger. If these constraints are violated, the communications aborts and the usage of the corresponding V2G service as well.

**Contributions.** This chapter presents the following main contributions:

- We define the threat model and analyze security requirements for secure provisioning, storage and usage of vehicle credentials for PnC and any V2G services based on the automated authentication in compliance with

[2] `https://www.iso.org/standard/77845.html`

[3] ISO 15118-20 [39] uses NIST P-521 and Curve448, which are not supported at least in the original specification.

[4] `https://github.com/tpm2-software`

ISO 15118. In order to provide a comprehensive security solution, we consider various strategies for generation and provisioning of the contract credential proposed in ISO 15118-2 [38], ISO/DIS 15118-20[5] [183], and the application guideline VDE-AR 2802 [78]: (i) contract keypairs and contract certificates are created in the eMSP's backend on the vehicle's request during a V2G session and delivered to the vehicle immediately; (ii) contract keypairs and contract certificate are created in the eMSP's backend in advance and stored in the Global Certificate Store (GCS) to be installed into the EV when it sends the request during a V2G session; (iii) contract keypairs are created in the vehicle and only the contract certificate is issued by the eMSP either on request during a V2G session and/or in advance as described above in (ii).

- We propose TrustEV, a novel security architecture for secure provisioning, storage, and usage of PnC credentials in an EV equipped with a HSM, by describing components for EVCC, OEM and eMSP and ISO 15118 protocol extensions for TrustEV support. Private keys of contract credentials can only be accessible in the protected environment of the HSM and can only be used if the provided access policy is met.
- We develop HIP, an architecture design that extends security properties of TrustEV with the secure and verifiable generation of all cryptographic keys used in V2G context in the vehicle's HSM and the trustworthy enrollment of credentials using these keys at an eMSP. The process assures that the cryptographic keys of the used contract credential are protected by the same HSM under the same policy as the OEM provisioning credential. HIP thus addresses the shortcomings of the current standard regarding credential management, where cryptographic keys are generated outside of the vehicle's secure environment.
- We introduce HIP 2.0, a further security extension building on HIP to provide the support of standard certificate issuance procedures using Certificate Signing Requests (CSRs) [184] and out-of-band credential provisioning via the application guideline VDE-AR 2802. HIP 2.0 improves HIP by enabling easy integration into existing backend infrastructures and processes and asynchronous creation and distribution of credentials to reduce performance bottlenecks.
- We investigate a concept for integrating and using TPM 2.0 features during secure provisioning of contract credentials compliant with ISO 15118-2, ISO 15118-20, and VDE-AR 2802. In particular, we address the ease of integration into the existing protocol as well as assuring backward compatibility to cover legacy systems.
- We analyze and discuss the security properties of the proposed protocol extensions considering strong adversaries.
- The main challenge was to develop the security extensions to ISO 15118 in a way that they provide the required protection level and stay conform with the standard. We evaluate several performance aspects of the proposed architectures using a Proof-of-Concept (PoC) with a hardware TPM 2.0 and demonstrate the compatibility of the solutions with the timing constraints of ISO 15118. With respect to the limitations on data sizes and data structures, we analyze the changes necessary for TrustEV, HIP and HIP 2.0 and discuss whether it is feasible to integrate the proposed security features into current implementations without effecting the core functionality. Message sequencing remains unchanged.

This chapter is organized as follows: We define our system model in Section 5.1 and the corresponding threat model Section 5.2. Then in Section 5.3 we derive security and functional requirements to be met by a viable solution. We address each provisioning scenario in a separate section and present the proposed security designs and their evaluation accordingly, i.e., TrustEV in Section 5.4, HIP in Section 5.5, and HIP 2.0 in Section 5.6. Related work is discussed in Section 5.7. We conclude the chapter in Section 5.8.

## 5.1. System Model

Our system model for PnC credential provisioning adopts actors and processes from the ISO 15118 standard series [37, 38, 39] and the application guideline VDE-AR 2802 [78]. We consider the EVCC installed in a vehicle and the SECC installed in a Charge Point (CP). The vehicle uses the EVCC to communicate with the SECC via ISO 15118 using the PnC authentication mode. The EVCC establishes a V2G session with the SECC, manages PnC credentials, and is

[5] The Draft International Standard (DIS) version was the publicly available version of the new edition of the standard at the moment of the publication [7, 8]. The provisioning process remained unchanged in the final Final Draft International Standard (FDIS) version [39], thus, the proposed approach remains applicable after adapting the cipher suits.

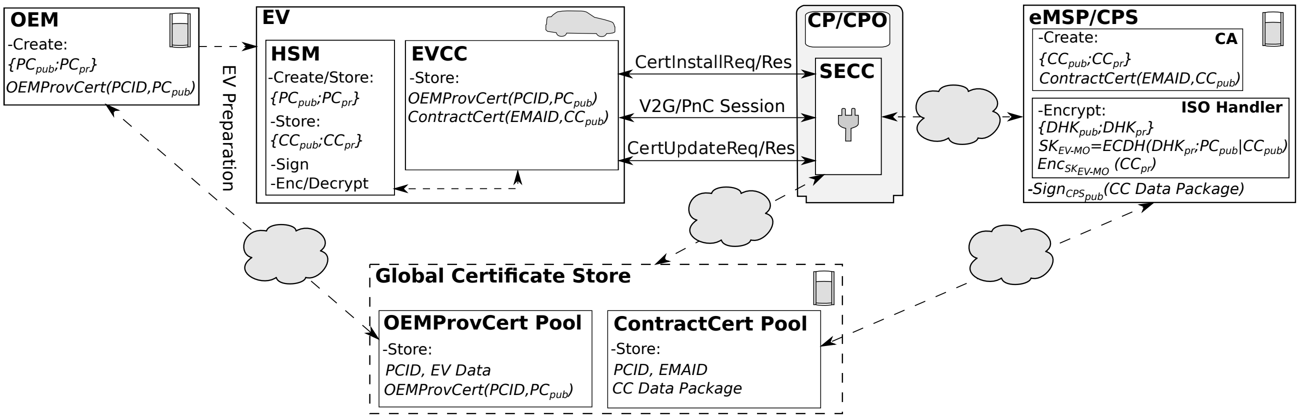


Figure 5.1: System Model for PnC Credentials Provisioning

the endpoint of EV authentication. The SECC enables data exchange between the EVCC and the backend systems of Secondary Actors (SAs) as well as verification of the vehicle's identity and authorization to use charging or other V2G services. Which SAs participate in provisioning processes depends on the supported provisioning strategy. The vehicle's OEM, the eMSP responsible for contract credentials, and the Certificate Provisioning Service (CPS) enabling delivery of the issued credentials to the EVCC are always involved. Within the eMSP, we distinguish the Certificate Authority (CA) issuing credentials and the ISO 15118 handler encrypting them for the recipient EVCCs. When contract credentials are delivered out-of-band, according to the VDE-AR 2802, the Global Certificate Store (GCS) consisting of the OEM provisioning and contract certificate pools for the intermediate storage of credential information is employed beside these actors. In addition, the EVCC can use an HSM installed in the vehicle for security services, such as secure key generation and storage.

Figure 5.1 shows our system model, where solid lines mark the components within the scope of ISO 15118. Dashed lines depict the out-of-band delivery use case and communications with SAs, which are required but not specified in detail by the standard.

In compliance with ISO 15118, the EVCC possesses the OEM provisioning credential and contract credential(s)[6]. The OEM provisioning credential consists of the keypair $\{PC_{pub}, PC_{pr}\}$ and X.509v3 *OEM provisioning certificate* $Cert_{PC}$ that binds the EVCC's long-term identity Provisioning Certificate Identifier (PCID) and $PC_{pub}$. This credential is used to authenticate the EVCC while installing or updating contract credential(s) during a V2G session [38, 39].

The contract credential consists of the keypair $\{CC_{pub}, CC_{pr}\}$ and X.509v3 *contract certificate* that binds $Cert_{CC}$ to a unique E-Mobility Account Identifier (EMAID) linked to an e-mobility service contract of the vehicle's owner. This credential is necessary for PnC and is used by the EVCC during automated authentication for contract-based charging. We consider several strategies for generation and provisioning of this credential: (i) contract keys and contract certificate are created in the eMSP's backend on the EVCC's request during a V2G session, as defined in ISO 15118 [38, 39]; (ii) contract keys and contract certificate are created in the eMSP's backend in advance and stored in the Global Certificate Store to be installed into the EV after the respective request is sent during a V2G session, as defined in VDE-AR 2802 [78]; (iii) contract keys are created in the vehicle and only the contract public key certificate is issued by the eMSP on request, in contrast to ISO 15118 and VDE-AR 2802. Necessary steps of the corresponding provisioning process are described in detail below.

With respect to credential provisioning, we also consider the identities of eMSP and the CPS. The eMSP holds the public key credential $\{MO_{pub}, MO_{pr}\}$ required by SAs to validate issued contract certificates. In our system model, we also adopt the CPS with the credential $\{CPS_{pub}, CPS_{pr}\}$ in its original role of a trusted third party that validates contract credentials provided to the EVCC by a potentially unknown eMSP and confirms their correctness and authenticity (cf. Section 7.9.2.5 in [38]). In some cases, the roles of eMSP and CPS can be assumed by the same SA. We use this fact to simplify our model. Similar to ISO 15118, we abstract communication paths between the SECC and the eMSP under the assumption of secure data transfer.

[6] The recent edition ISO/FDIS 15118-20 additionally introduces a vehicle credential used for mutual authentication in TLS, both in the PnC and External Identification Means (EIM) authentication modes. Though the vehicle credential is necessary for the V2G communication (if TLS fails, contract-based charging is not possible) and thus can potentially be misused by CPOs to prevent certain vehicles from accessing PnC services, we do not consider it as a PnC credential.

The provisioning processes specified in ISO 15118-2, ISO 15118-20 and VDE-AR 2802 involve multiple largely independent actors and their success strongly depends on coordinated execution of their required steps by these actors. In order to ensure that our solutions remain conform to the established standards, we summarize these steps below as part of our system model.

### 5.1.1. Credential Provisioning Process via ISO 15118

ISO 15118 allows the EVCC to request its contract credential(s) over the same communication channel as it already uses for (dis)charging. This can be helpful even for modern connected vehicles when they travel in areas with poor or no coverage. In order to enable this useful feature, the following steps need to be performed, partially, before the ISO 15118 communication begins.

**EV Preparation.** During manufacturing, before the vehicle is delivered to its future user, the OEM generates the EV's unique long-term identity for credential provisioning PCID and the corresponding OEM provisioning credential.

**Generation of Contract Credentials.** In order to register the EV at an eMSP for using PnC and conclude the service contract, the EV user presents the OEM provisioning certificate to the eMSP. Based on the information from this certificate, the eMSP issues the contract credential allowing the vehicle to use contract-based charging. As per ISO 15118, the eMSP encrypts the private contract key to protect its confidentiality during transport to the EVCC over potentially insecure channel via intermediaries (see [38], Section 7.9.2.4.3). The encryption uses the session key $SK_{EV-MO}$ derived using ephemeral-static Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol, where the ephemeral part $\{DHK_{pub}; DHK_{pr}\}$ is newly generated by the eMSP and the static part is taken from one of the EVCC's identity credentials. In the case of ISO 15118-2, this part is either the $PC_{pub}$ from the OEM provisioning certificate for the initial enrollment, or $CC_{pub}$ from the contract certificate for update. In ISO 15118-20, certificate update is deprecated and only $PC_{pub}$ is applied. The contract credential data package including contract credentials, the eMSP certificate chain, and $DHK_{pub}$ is signed by the CPS.

**Installation and Update of Contract Credentials.** The contract credential is installed into the EV when the EVCC requests PnC at a Charge Point, but does not possess a valid credential for authentication. For this purpose, the EVCC sends a request message to the SECC of the Charge Point $CertInstallReq$ or $CertUpdateReq$, for installing or updating the contract credential, respectively. The installation request is authenticated with $PC_{pr}$, while the update request with $CC_{pr}^{old}$. The SECC forwards the request to the eMSP's backend system. If the response with the contract credential data package is received from the backend, the SECC delivers it to the vehicle. The EVCC verifies the authenticity of the provided data and uses the attached $DHK_{pub}$ to derive the ECDH session key and decrypt its new $CC_{pr}$. Finally, the credentials are stored on the EVCC and can be used for PnC.

**Contract Credential Usage.** The EV uses the contract credentials for PnC and, optionally, to sign metering confirmations sent by the SECC during charging. In order to start charging, the EVCC provides its EMAID and contract certificate $Cert_{CC}$ to the SECC. If the SECC accepts the provided credential, it initiates the challenge-response authentication protocol and sends the EV a random challenge. The EV signs this challenge with the private key $CC_{pr}$ and sends the challenge and the signature results back to the SECC (cf. $AuthorizationReq$ message in [38]). If the authentication succeeds, the SECC activates charging. The EVCC may periodically receive and sign meter readings with its private key $CC_{pr}$ to confirm the status of the charging session.

### 5.1.2. Credential Provisioning Process via VDE-AR 2802

In addition to the standard ISO 15118 communication, the application guideline VDE-AR 2802 enables intermediate storage of certificates and auxiliary information, so that generation and provisioning processes can be carried out independently by the SAs and time-consuming operations can be completed before the vehicle arrives at a Charge Point.

**EV Preparation.** In addition to the previously described steps, after generating the OEM provisioning credential for the vehicle, the OEM stores its public-key certificate under the respective PCID in the the Global Certificate Store (cf. Figure 5.1).

**Issuance of Contract Credentials.** When the eMSP receives the vehicle's PCID from the user, it extracts all information necessary for generating contract credentials for this EV directly from the GCS. Having generated the contract credential, with the help of its CA and ISO 15118 handler, the eMSP transfers it within the contract credential data package to the CPS for signing. After being signed, the package is stored in the GCS under both identifiers, the EMAID and the PCID.

**Contract Credential Installation.** In contrast to the standard ISO 15118 installation process, when the contract credential is generated "on-the-fly" with the data provided in the installation or update request from an ongoing PnC session, VDE-AR 2802 uses V2G communication protocols only as one of delivery methods. Upon receiving the request from the EVCC, the SECC forwards it to the CPS, as usual. The CPS extracts the EVCC's identifier – PCID, or EMAID if update is performed – from the request and downloads all contract credential data packages available for this identifier from the GCS. The result is returned to the SECC, which packs these data packages into installation or update response messages sent to the EVCC. Finally, the EVCC extracts and installs contract credential(s) from the responses provided by the SECC, after successful verification of their authenticity, and can use them as usual.

### 5.1.3. Assumptions

Based on the security requirements specified in ISO 15118-2, ISO 15118-20 and VDE-AR 2802, we assume that strong and well-established cryptography is used to protect sensitive provisioning data against disclosure or theft by intermediate actors. As TLS channel is mandatory for all PnC communications, data transfer between the EVCC and the SECC can also be assumed to be secure. With OCPP 2.0's TLS profiles [130] in mind, we assume that secure communication channel is available between a Charge Point and the backend systems, too. Though ISO 15118 mentions that $PC_{pr}$ and $CC_{pr}$ are sensitive data and may need additional protection, it does not specify how these private keys should be stored on the EVCC. Thus, current *Root of Trust* for PnC is the complete firmware of the EVCC supposed to securely store and use the keys.

Following the trust relationship realized by means of the V2G PKI described in ISO 15118-2 and ISO 15118-20, we additionally assume the SAs including OEM, eMSP, and CPS to be trustworthy and provide non-corrupted data. If the OEM or the eMSP generate cryptographic keys in their respective backends, we assume that necessary security measures are assured. Otherwise, EV users can claim that their contract credentials have been leaked by the service provider and refuse to pay the bills.

## 5.2. Threat Model

In our threat model, we consider an adversary who aims to extract, duplicate or copy PnC credentials of an EV, i.e., OEM provisioning and contract certificate(s) together with their private cryptographic keys, and to use these credentials for identity-based attacks against the PnC service, service provider(s) or the EV user. Getting in possession of PnC credentials allows the adversary to impersonate the vehicle owned by a legitimate PnC service customer and charge other EVs or use value-added services at the expense of this customer. The adversary is also able to invalidate the customer's contract credentials and to disturb the service usage for the victim at will simply by requesting a new credential at a Charge Point. Notably, it is hard for the eMSP to detect the theft or misuse of customers' PnC credentials or to single out potential fraudsters, due to normally prolonged billing periods (minimum, one calendar month) and because Charge Detail Records (CDRs) provided by CPOs lack such information as EVCCID/MAC that could help identify the vehicle that uses the stolen credentials. Apart from causing financial harm (or gaining profit), the adversary can use the trusted vehicle's identity to access the V2G infrastructure in an unauthorized manner and to destabilize the local power grid through uncoordinated or unpredictable charging behavior.

The potential attack surface of PnC credential provisioning according to the system model presented in Section 5.1 includes the EVCC, the SECC, several backend systems of SAs, and their network connections. Depending on which provisioning option the eMSP chooses to support, beside eMSP and CPS backends that are always present, the GCS can also be added to the composition of the attack surface.

However, ISO 15118 and backend communication standards define security measures for PnC restricting potential adversaries (cf. Section 5.1.3). This allows us to assume that the respective communications are secure under the Dolev-Yao adversary model [128] and to consider attacks via network channel and any attacks on the deployed cryptography out of scope for this work. Moreover, the SECC deals only with signed and encrypted contract private keys during provisioning and therefore even in the case of compromise cannot be used by the adversary to hijack contract credentials, but only to disrupt their delivery.

Under these premises, we consider a powerful local adversary, *car hacker*, who has a full physical access to the EV and can modify or replace its components, install manipulated firmware, and extract or copy any stored data, except when these data are tamper-protected. Possible attack vectors that can be exploited by the car hacker include offline attacks against EVCC's flash memory or storage as well as runtime attacks against Random Access Memory (RAM). Also, the car hacker may relay any authorization or certificate requests to the EVCC to make the controller sign them for the adversary. We consider side-channel attacks out of scope, because they are extremely costly compared to the previously described methods and do not scale.

We also consider the second type of adversaries, a *backend hacker*, who can compromise a backend system responsible for the issue or the delivery of PnC credentials and get access to the respective private keys unless the responsible SA ensures the adequate (certifiable) protection.

The car hacker and backend hacker can belong to one of the following categories:

**Evil mechanic.** An employee of an OEM's repair shop who performs EV maintenance. The "evil mechanic" first introduced in [185] has full physical access to the vehicle and the required equipment to steal PnC credentials. Again, these attacks are hard to detect and prove without additional safeguards, which puts the evil mechanic on the safe side.

**EV owner.** Depending on the charging conditions in the contract, EV owners may gain monetary benefits from sharing their e-mobility service contract with the family or friends or among own vehicles, e.g., if the eMSP offers a flat rate tariff. The owner has a full physical access to the EV and can simply copy the credentials or hire an evil mechanic to carry out the attack instead.

**EV driver.** A customer of a car sharing service, who has a temporary access to an EV and may wish to copy its PnC credentials for personal use.

**Cyber criminal.** A technically savvy criminal who aims to steal PnC credentials for selling (or ransom) or for gaining access to the V2G infrastructure and launch attacks against the local electric grid. The criminal can get access to an EV while it is parked or charging[7], or alternatively, use scrapped EVCCs. This adversary can also search leaked databases for stored credentials or even carry out attacks against backend systems using known vulnerabilities and publicly available exploits.

**Unfair competitor.** In order to facilitate their business, an OEM or a service provider may wish to purposefully put their competitor at a disadvantage by creating bad publicity around crafted cases of incorrect customer or B2B billing and/or around security issues regarding a particular vehicle model or type of service. Companies usually dispose of sufficient means and resources to arrange or sponsor car hacker's or backend hacker's attacks.

**Malicious insider.** Malicious insiders are employees of the OEM or one of the SAs involved into PnC credential provisioning. They also wish to harm their company or gain profit with copied or falsified credentials.

As the V2G infrastructure connects transportation with energy supply, both recognized critical infrastructures, nation state type adversaries may also be interested in gaining access to PnC credentials to be able to launch coordinated attacks at will [186]. Though currently the number of registered EVs might not yet be sufficient to cause country-wide disruptions [46], but it may suffice to produce local blackouts.

## 5.3. Requirements

With our extension(s) of the ISO 15118 protocol, we aim to increase the Level of Assurance (LoA) (cf. ISO 29115 [187]) for EVCC authentication. Currently, even if an actor in the V2G ecosystem can successfully verify the identity provided by an EVCC, e.g., to use PnC or any other V2G service, it cannot trust that this identity actually belongs to this EVCC. Especially, the V2G service billing and energy feedback but also Value Added Services (VAS) work on this assumption and, therefore, are vulnerable to malicious actions performed on behalf of entities that can authenticate themselves as trustworthy EVCCs.

[7] https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob/

In this section, we define requirements for an EV-centric solution necessary to achieve the above goal from security and functional standpoint. According to our threat model, we assume that appropriate security mechanisms are deployed to secure backend communications and backend systems of the respective Secondary Actors (SAs).

### 5.3.1. Security Requirements

Our solution should prevent the adversary described above (cf. Section 5.2) from accessing and/or using PnC (or more generic V2G) credentials that represent the identity of a particular vehicle within the V2G ecosystem and, thus, mitigate the attacks on PnC authentication. Since V2G applications are safety-critical, they require the highest LoA4 in accordance to ISO/IEC 29115 [187]. Thereby, the LoA describes the risk related to the event that an entity successfully authenticates itself using an identity that belongs to other entity. The higher the LoA is, the higher is the risk associated with this event. The highest LoA4 is assigned if such errors can have safety impacts, at least in one of possible threat scenarios. This is the case with the V2G authentication, as we discuss in more detail in Chapter 6. For LoA4, ISO 29115 requires the usage of HSMs for the storage of cryptographic keys as well as the cryptographic protection of any data used for the authentication [187]. The second part is partially fulfilled by ISO 15118 when a TLS-based secure channel is used for PnC. Otherwise, the only encrypted piece of information is the private contract key, which has to be decrypted on the EVCC to be used during authentication. Therefore, we propose to create, when possible, and store OEM provisioning and contract credentials in the HSM of the customer's EV. If PnC credentials never leave the protected environment of the vehicle, they are out of reach for backend hackers and are better protected against large-scale attacks, when multiple e-mobility customer accounts can be stolen at once.

On the downside, many eMSPs consider contract credentials their property and wish to stay in control of how these credentials are created, even if the associated provisioning process is complex and potentially error-prone. Such eMSPs also need to ensure the adequate protection against the backend hacker (cf. Section 5.2) and to provide the necessary evidence that customers' accounts are safe. If they choose to accept self-created credentials, the credential generation in the vehicle has to meet security standards expected by the eMSP and mandated by regulations. Moreover, the eMSP may demand the EV to prove that the contract credentials are generated and stored securely.

We summarize the above considerations in the following *security requirements* to be fulfilled by a suitable solution.

$RS_1$ *Secure key storage:* Private cryptographic keys of PnC credentials shall be stored in a protected and secure environment to prevent their leakage.

$RS_2$ *Secure cryptographic operations:* In order to ensure that EVCC's cryptographic keys cannot be leaked during usage, an environment for secure execution of cryptographic algorithms physically separated from other hardware components of the EVCC shall be provided.

$RS_3$ *Key usage authorization:* In order to prevent arbitrary access to private cryptographic keys by an unauthorized service on the EVCC's processor or by a manipulated firmware, access to these keys shall be limited to authenticated and trusted components.

$RS_4$ *Secure key provisioning:* Private cryptographic keys of PnC credentials shall be generated and deployed in such a way that an adversary cannot gain access to these keys during their transport or storage.

$RS_5$ *Cryptographic agility:* The solution shall provide a method to update or replace outdated or insecure cryptographic primitives and algorithms. It is worth mentioning that ISO 15118-2 lacks cryptographic agility, i.e., this first edition specifies an exact cryptographic algorithm to be used for a particular task as part of the standard requirements. This decision is motivated by interoperability reasons, limited controller resources and that chosen algorithms and key lengths are considered secure at the moment (of specification). The new edition ISO 15118-20 partially addresses the need of long-term security for EVs with the expected lifespan being more than 12 years (even up to 40 years according to the standard) and provides the capability to exchange a vulnerable default cryptographic algorithm with an alternative one. The only disadvantage is that both algorithms are Elliptic Curve Cryptography (ECC)-based and cannot protect the V2G infrastructure in the Post Quantum (PQ) era.

In the case where the eMSP agrees to accept the credentials generated on the vehicle itself or its EVCC, the following *additional security requirements* apply:

$RS_6$ *Secure key generation:* Cryptographic keys including those for PnC credentials shall be generated within a protected and secure environment within the EV they are destined for. All required random numbers shall also be generated within this environment. This environment shall be physically separated from the hardware components.

$RS_7$ *Trustworthy credential enrollment:* The EVCC shall only be able to enroll a self-generated contract credential at an eMSP, i.e., obtain a contract certificate for the public key of this credential from the eMSP that it can use for authentication, if the private key was generated in a protected and secure environment of its vehicle and with the expected predefined parameters.

### 5.3.2. Functional Requirements

We aim for a usable solution that can be easily adopted in the existing V2G infrastructure. Such solution needs to be compliant with the established standards and operating practices of OEMs and service providers, whenever it is possible without undermining security, and to incur minimal overhead during its deployment and usage.

Therefore, in order to ensure high acceptance and easy deployment, the solution should address both the current and upcoming edition of the ISO 15118 standard as well as various credential provisioning strategies that can be adopted by eMSPs (cf. Chapter 2, It should be noted that it is not required for the solution to work with both protocol editions at the same time, because these editions are incompatible. When establishing a V2G communication session, the supported protocol version is negotiated between the EVCC and the SECC and cannot be changed until the (dis)charging process is finished.

We reflect these design goals in the following *functional requirements*:

$RF_1$ *Minimal overhead.* The solution shall keep any additional computational and communication overhead possibly low if compared with the execution of the original protocol flow defined in ISO 15118 [38, 183].

$RF_2$ *Conformance to the ISO 15118 protocol specifications.* The solution shall preserve the standardized message sequences of the respective protocol version for EVCC, SECC, and the secondary actors eMSP and CPS and shall meet the specified time and data size constraints. Whenever possible the specified message formats shall be preserved (otherwise, the message can be rejected by an intermediate or receiving party); any newly introduced elements shall be optional.

$RF_3$ *Backward compatibility.* The system shall not impede the operation of EVCC, SECC, or CPS and eMSP that comply with ISO 15118, even if any of these actors does not support the proposed security extension.

In the case where the eMSP cannot deliver contract credentials during the charging session, e.g., because the predefined time constraints (5 seconds) cannot be met, and opts for the alternative provisioning method specified in the application guideline VDE-AR 2802 [78], the following *additional functionality* is required:

$RF_4$ *Conformance to VDE-AR 2802:* The solution shall permit out-of-band contract credential provisioning according to the application guideline VDE-AR 2802 [78], which uses an intermediate storage for credentials shared by OEM, eMSP, CPS, and Charge Point/CPO and the supplementary information exchange between the corresponding backend systems.

$RF_5$ *Conformance to standard procedures for CAs:* The initial issue and the renewal of contract certificates shall follow the standard procedure of requesting and issuing PKIX certificates using CSRs as specified in RFC 5280 [184].

In the following sections, we describe three security architectures aiming to address various aspects of PnC credential provisioning, and discuss how each of them fulfills the security and functional requirements specified above.

## 5.4. TrustEV: Trustworthy Electric Vehicle Charging and Billing

In this section, we introduce our novel TrustEV security architecture for secure provisioning of PnC credentials in accordance with the standard provisioning strategy defined in ISO 15118, where an eMSP is responsible for prompt generation and delivery of a new contract credential on request from its customer's EV during an active PnC session. The primary goal of TrustEV is to protect PnC credentials from the car hacker (cf. Section 5.2), under the assumption that eMSPs following the original provisioning process ensure their protection in the backend. In order to satisfy the respective security requirements (cf. Section 5.3.1) and achieve the required LoA, the TrustEV functionality relies on a certified HSM for secure storage of private keys and cryptographic operations on the vehicle's EVCC. As the EVCC receives an encrypted private contract key from the eMSP, it needs to decrypt it before usage, without exposing this key to the car hacker. For this purpose, TrustEV allows the OEM and eMSPs of an EV to collaborate and exchange security information necessary for secure provisioning of contract credentials and makes it possible for private keys to be directly imported into the target HSM, so that they cannot be decrypted and used outside of its secure environment. But if car hackers cannot copy the key, they can install an exploit redirecting signing requests to the HSM storing the key. Therefore, TrustEV also provides a method to restrict access to private PnC keys using specially crafted policies in such a way that a manipulated EVCC cannot misuse them. To provide a solution ready for integration into an international standard and to enable a reproducible benchmark for PoC, we adapt an openly standardized TPM 2.0 to define the import format and HSM Application Programming Interface (API) for TrustEV. This is also an advantage for the companies like Volkswagen planning to use TPMs in their cars [132].

In order to achieve the design goals determined in Section 5.3.2, the main challenge is to develop TrustEV in such a way that it provides the required protection level and remains conform to the ISO 15118 standard. Moreover, the TrustEV protocol extensions are backward compatible, i.e, if an actor does not support this security feature, it can proceed with credential provisioning as usual. This is important because the V2G infrastructure is already being deployed and not all components can be easily upgraded. In our case, adaptations are required in the initial preparation of the EV performed by the OEM, the generation of a contract credential and encryption of its private key, OEM provisioning and contract certificate profiles, and the charging process to make use of the TPM. Except of the encrypted private key and certificate profiles, these changes are not relevant for ISO 15118 message format and sequencing. In order to show that TrustEV does not have any impact on core functionalities, we evaluate its security and performance using a proof-of-concept implementation.

In summary, we make the following main contributions:

- We present TrustEV, a novel security architecture design providing secure provisioning, storage, and usage of PnC credentials in an EV equipped with a HSM, by describing components for EVCC, OEM and eMSP and ISO 15118 protocol extensions for TrustEV support.
- We provide a concept for integrating and using TPM 2.0 features during secure provisioning of contract credentials compliant with ISO 15118-2 and ISO 15118-20. In particular, we address the ease of integration into the existing protocol as well as assuring backward compatibility.
- We analyze and discuss the security properties of TrustEV considering strong adversaries with control over the EVCC and communication channels.
- We evaluate several performance aspects of TrustEV and demonstrate the compatibility of the solution with the timing limitations of ISO 15118.
- We provide the input to the ISO/FDIS 15118-20 [39] and successfully integrate TrustEV into the upcoming edition to resolve incompatibilities identified during PoC evaluation.

This section is organized as follows: We introduce the components of the TrustEV security architecture in Section 5.4.1 and describe their integration into the ISO 15118 protocol flow in Section 5.4.2. We provide an informal discussion of the security properties of TrustEV and fulfillment of the previously defined security requirements in Section 5.4.3. The proof-of-concept implementation and functional evaluation are described in Section 5.4.5. As contrary to our expectations, significant changes in cryptographic algorithms have been introduced between preFDIS version [183] available at the moment of our publication [1] and the final version of ISO/FDIS 15118-20 [39], in Section 5.4.6 we shortly describe the necessary changes to TrustEV parameters and discuss what additional changes we made to the specification to support TrustEV functionality. We summarize the results of our study in Section 5.4.7.

### 5.4.1. TrustEV Components

An overview of the TrustEV security architecture is given in Figure 5.2, where the dark gray boxes represent new TrustEV components extending the system model described in Section 5.1.

The `TrustEV-OEM` component in the OEM backend system is used by the OEM during EV manufacturing to create the required cryptographic keys the vehicle's HSM and OEM provisioning certificate. The keys created by the OEM are the OEM provisioning keypair $\{PC_{pub}; PC_{pr}\}$ and the newly introduced asymmetric *Storage Root Key (SRK)*. The SRK and all additional parameters required to securely store encrypted private contract keys in the HSM are transported within the TrustEV extension `TrustEVExt` of the OEM provisioning certificate, as explained later in this section.

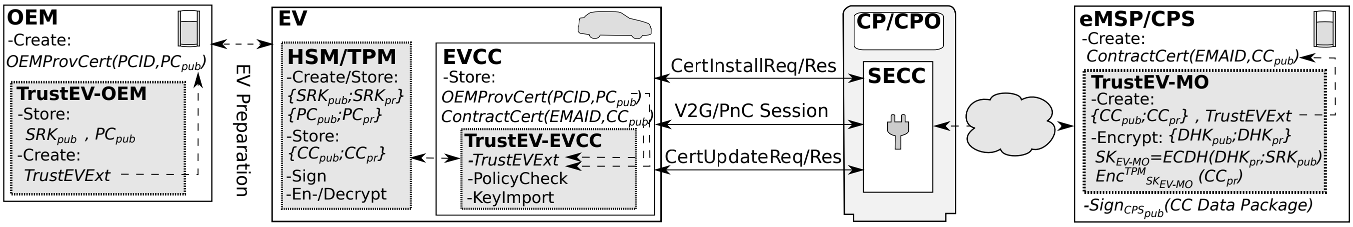


Figure 5.2: TrustEV Security Architecture

The TrustEV-EVCC component in the vehicle is responsible for the communication with the HSM providing security services to the EVCC. In this work, we use the standardized hardware TPM 2.0 [114] for storing sensitive PnC credentials, executing cryptographic operations, and implementing measured boot. Furthermore, the TPM is used to seal PnC credentials to the EVCC's firmware state or other access policy that further restricts the usage of these credentials. If OEM defines any such policy, it is also included into the TrustEV extension.

The TrustEV-MO component in the eMSP backend determines by means of the TrustEV extension in the provided OEM provisioning certificate whether the EV requesting the contract certificate installation supports the TrustEV functionality. If this is the case, TrustEV-MO creates and encrypts the contract key in such a way that it is sealed to the OEM's policy and can be imported directly into the vehicle's HSM without being first decrypted in the memory. If a version of ISO 15118-2 is executed, this component also adds the TrustEV certificate extension TrustEVExt to the respective contract certificate for future updates.

In summary, the TrustEV architecture makes changes in the EVCC by integrating and using an HSM/TPM for securing PnC credentials; in the OEM's EV preparation process; in the generation and encryption of contract keys and certificates by the eMSP, and in the ISO 15118 certificate profiles by adding a new extension. In the following, we describe the proposed key hierarchy for the protected storage and the TrustEV certificate extension in detail.

**TrustEV Key Hierarchy.** For secure storage and import of cryptographic keys with TrustEV, we adopt the TPM's protected storage hierarchy (see [114], Chapter 23). We also adopt the term Storage Root Key (SRK) to refer to the root of the TrustEV key hierarchy and the parent for all keys participating in PnC credential provisioning. In our case, it may but does not have to be a primary key of the TPM. The SRK is an asymmetric key generated by the TPM and is used to ensure integrity and confidentiality of the keys stored under it, through encryption and policy validation [113]. The SRK cannot be used for general-purpose cryptography to avoid attacks, where an adversary tricks the TPM to output a key decrypted with the SRK to an unprotected storage, and, thus, leak this key. In addition, the SRK works only with specific commands and on strictly formatted objects, which can be unambiguously recognized as keys and treated accordingly. This is also the case with direct import of contract keys into the TPM, which need to have a specified format, as we discuss later. For this reason, the OEM provisioning key used for signing installation requests and decrypting private contract keys cannot serve as a parent for imported keys.

If an encrypted key needs to be imported into the TPM, it has to be encrypted with its future parent (cf. [114], Chapter 23.3.2). Therefore, the contract key encryption method at the eMSP has to use the public part of the pair $\{SRK_{pub}; SRK_{pr}\}$, instead of the public OEM provisioning key $PC_{pub}$. In order to securely transfer the alternative static public key for ECDH to the eMSP, $SRK_{pub}$ is included in the TrustEV X.509v3 extension of the OEM provisioning certificate by the OEM.

The resulting TrustEV key hierarchy is shown in Figure 5.3. The dashed line shows the contract key decrypted on the EVCC for backward compatibility, before storing it in the TPM. In the squared brackets, we list TPM attributes for the keys in the TrustEV key hierarchy reflecting the requirements of both ISO 15118 [38] and the TPM specification [114]. The key attributes *SIGN* and *DECRYPT* refer to the intended key usage, *RESTRICTED* limits the key usage to specific commands and object formats, *FIXEDTPM* and *FIXEDPARENT* prohibit exporting the key out of the TPM or move it within the key hierarchy, and, finally, *USERWITHAUTH* indicates that the key usage needs to be authorized via a password or a policy assertion [114].

**TrustEV Certificate Extension.** Similar to ISO 15118, we use the OEM provisioning certificate signed by the OEM to provide in an authenticated manner the EVCC's TPM public key and policy information required by TrustEV. These inputs can alternatively be provided via a trusted channel between the OEM and eMSP backends. However, we cannot

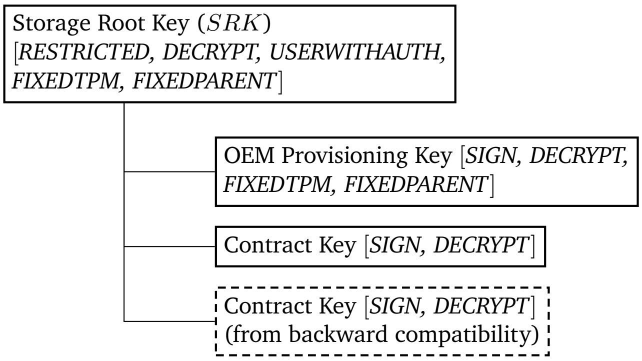

Storage Root Key ($SRK$)
[RESTRICTED, DECRYPT, USERWITHAUTH, FIXEDTPM, FIXEDPARENT]
OEM Provisioning Key [SIGN, DECRYPT, FIXEDTPM, FIXEDPARENT]
Contract Key [SIGN, DECRYPT]
Contract Key [SIGN, DECRYPT]
(from backward compatibility)

Figure 5.3: TrustEV Key Hierarchy

assume that such channel exists for ISO 15118, because all data required for credential provisioning need to be sent in the corresponding request. The EVCC cannot be trusted to announce a different encryption key for itself under our threat model (cf. Section 5.2).

Figure 5.4 shows an example OEM provisioning certificate with the TrustEV extension consisting of two octet strings: the public Storage Root Key (SRK) and the digest of the OEM policy used to seal OEM provisioning and contract keys. As mentioned before, an eMSP needs this information to encrypt a new private contract key in a way suitable for the direct TPM import. The TrustEV extension is included as non-critical$^{nc}$ into OEM provisioning certificates and, for ISO 15118-2, in contract certificates. Thus, it can be ignored by any actor that does not support TrustEV to avoid incompatibility. There are no further changes made to the certificate profiles specified in ISO 15118 (cf. [38], Annex F).

| OEM Provisioning Certificate | | |
|---|---|---|
| Version: | | X.509v3 (0x2) |
| Serial Number: | | 12345 (0x3039) |
| Signature Algorithm: | | ecdsa-with-SHA256 |
| Issuer: | | CN=OEMSubCA2, O=ISO, C=US |
| Validity | Not Before: | Nov 15 08:40:32 2020 GMT |
| | Not After: | Nov 15 08:40:32 2022 GMT |
| Subject: | | CN=OEMProvCert, O=ISO, DC=OEM |
| Subject Public Key Info | Public Key: | OCTET STRING |
| | Algorithm: | id-ecPublicKey |
| | Parameters: | namedCurve secp256r1 |
| X509v3 Extensions | Basic Constraints:$^{c}$ | CA:FALSE |
| | Key Usage:$^{c}$ | Digital Signature, Key Agreement |
| | Subject Key Identifier:$^{nc}$ | keyIdentifier (SHA-1) |
| | **TrustEV Extension:**$^{nc}$ | **TPM 2.0 EC Storage Root Key ($\text{SRK}_{\text{pub}}$)<br>512 bit OCTET STRING** |
| | | **TPM 2.0 SHA256 Policy Digest<br>256 bit OCTET STRING** |
| Signature | Algorithm: | ecdsa-with-SHA256 |
| | Value: | OCTET STRING |

Figure 5.4: Provisioning Certificate with TrustEV Extension

### 5.4.2. TrustEV Protocol Extensions

In order to integrate TrustEV into the ISO 15118 credential provisioning process (cf. Section 5.1.1), we define extensions to the original protocol, where our newly introduced components are engaged. We follow the previously discussed standard steps and describe necessary changes below.

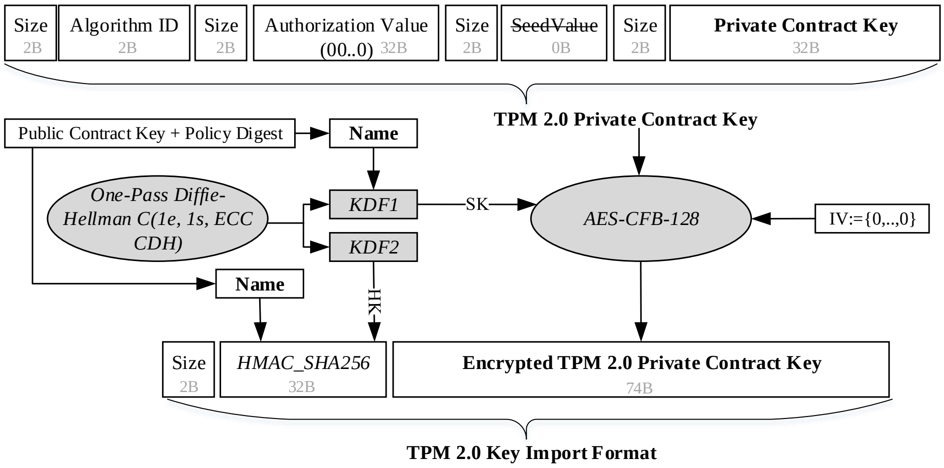


Figure 5.5: TPM 2.0 Key Encryption and Import Format

**EV Preparation.** Before creating the OEM provisioning credential for an EV, the OEM uses the vehicle's TPM to generate the root of the TrustEV key hierarchy $\{SRK_{pub}; SRK_{pr}\}$. Next, the OEM decides on a policy restricting the usage of PnC credentials, i.e., some predefined condition that needs to be satisfied to be able to access or, in TPM terminology, load their private keys. The enhanced authorization in TPM 2.0 [113] allows OEMs to implement a large variety of policies. For demonstration purposes, we use a simple assertion policy `TPM2_PolicyPCR` restricting the usage of keys in the TrustEV hierarchy to the software state registered in one or several Platform Configuration Registers (PCRs). This way, TPM can load these keys and use them for cryptographic operations only if the policy assertion is correct, i.e., the EVCC's software is in a known state. Afterwards, the TPM is triggered to generate the asymmetric OEM provisioning key $\{PC_{pub}; PC_{pr}\}$ being stored under the SRK (cf. Figure 5.3) and sealed using the chosen TPM policy. The public part $SRK_{pub}$ and the policy digest are included the TrustEV extension of the OEM provisioning certificate $Cert_{PC}$.

**Generation of Contract Credentials.** When we introduced the TrustEV key hierarchy, we mentioned that only specially formatted and encrypted keys can be directly imported into the HSM of our choice, i.e., TPM 2.0, without being first decrypted in the EVCC's memory. The format used by ISO 15118 does not meet the demands of the chosen HSM[8]. Therefore, TrustEV has to adopt an alternative format and encryption method for private contract keys compatible with the TPM functionality. All other actions performed before this step remain unchanged (cf. Section 5.1.1).

First, the eMSP verifies the received identity credential of the EVCC including the TrustEV certificate extension and stores $SRK_{pub}$ and the policy digest if it is valid. Then, `TrustEV-MO` generates a new contract credential using the cryptographic algorithm specified in ISO 15118, with only one difference when ISO 15118-2 is executed: the contract certificate receives the TrustEV extension with $SRK_{pub}$ and the policy digest necessary for credential updates (cf. [38] Section 8.4.3.10). To recall, the latest edition ISO 15118-20 discontinues credential updates with contract certificates and does not require any changes to contract certificates. Afterwards, the format necessary for the direct import into the target HSM is applied to the private key $CC_{pr}$ of the new contract credential. In the case of TPM 2.0, the private key receives a prefix indicated as *sensitiveArea* in Figure 5.8. This prefix includes an identifier of the cryptographic algorithm (i.e., ECC for ISO 15118 keys), optional authorization value (e.g., a password), and a seed value (not used in our use case).

Finally, the entire structure is encrypted using a method required by the HSM. In comparison to ISO 15118, there are three major changes necessary with the TPM 2.0 (see Figure 5.5):

1. instead of the public key of the EVCC's identity credential, i.e., $PC_{pub}$ or $CC_{pub}$, the TPM identity is used;

2. different encryption mode of the symmetric cipher (i.e., CFB); and

[8] Though ISO 15118 provides with XML a structured data representation, which allows any processing party to locate the encrypted key, the used method of integrity protection is susceptible for attacks. Instead of embedding the binding between the public and the private part of the key together with desired access restrictions into the key encryption, ISO 15118 uses a trusted third party that did not generate the key and is not able to validate it to authenticate the XML structure containing sensitive key data.

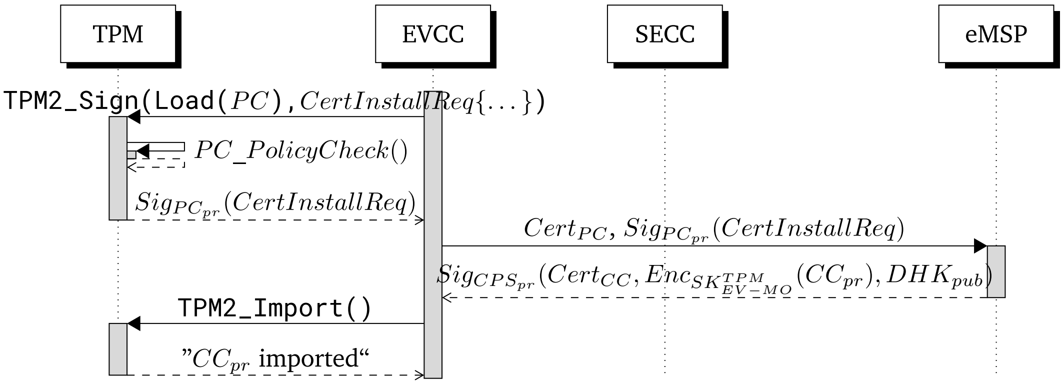

TPM
EVCC
SECC
eMSP
TPM2_Sign(Load($PC$),$CertInstallReq\{\ldots\}$)
$PC\_PolicyCheck()$
$Sig_{PC_{pr}}(CertInstallReq)$
$Cert_{PC}$, $Sig_{PC_{pr}}(CertInstallReq)$
$Sig_{CPS_{pr}}(Cert_{CC}, Enc_{SK^{TPM}_{EV-MO}}(CC_{pr}), DHK_{pub})$
TPM2_Import()
”$CC_{pr}$ imported“

Figure 5.6: Installation of Contract Credentials

3. an authentication value (only padding, because we use a policy instead of a password) is added to the encrypted key structure.

Thus, the symmetric session key $SK^{TPM}_{EV-MO}$ for encrypting $CC_{pr}$ is calculated using ephemeral-static ECDH key exchange with the static key $SRK_{pub}$. The policy digest and public key information are also included in this process to provide the necessary binding. After encrypting the private key, an HMAC is added to the resulting ciphertext according to the definition of ”Outer Duplication Wrapper“ (cf. [114], Chapter 23.3.2.3). Figure 5.8 shows the entire resulting structure for the direct TPM import.

Except of the already described changes in the certificate profile, no additional parameters are required, which allows for an easy integration into the current ISO 15118 protocol flow using the already available message formats.

**Installation and Update of Contract Credentials.** The EVCC with TrustEV executes the standard message sequence to install or update its contract credentials as shown in Figure 5.6. Since with TrustEV PnC keys are stored under HSM protection, `TrustEV-EVCC` needs first to call the corresponding function of its TPM 2.0 to perform a desired cryptographic operation, e.g., signing the installation request $CertInstallReq$ with the OEM provisioning key $PC_{pr}$. Before this key can be used the access policy, i.e., `TPM2_PolicyPCR` defined earlier by the OEM needs to be validated. If these operations succeed, the signed request $Sig_{PC_{pr}}(CertInstallReq)$ is returned. The EVCC sends this request together with the OEM provisioning certificate $Cert_{PC}$ using the current PnC session to the SECC, which forwards it to the eMSP. Its `TrustEV-MO` component generates the contract credentials as described above and sends the contract credential data package containing $Cert_{CC}$, $Enc_{SK^{TPM}_{EV-MO}}(CC_{pr})$, and $DHK_{pub}$ signed by the CPS back to the EVCC. Then, the encrypted private contract key $CC_{pr}$ is directly imported into the TPM by using the function `TPM2_Import()`, which uses $SRK_{pr}$ instead of $PC_{pr}$ for deriving the ECDH session key $SK^{TPM}_{EV-MO}$ and decrypting $CC_{pr}$ in the protected environment. In summary, only the systems aspects of the installation process are altered, the rest is carried out without changes.

The update process is similar to the installation process, with the update request $CertUpdateReq$ and current contract key $CC^{old}_{pr}$ used instead (cf. Section 5.1.1).

**Contract Credentials Usage.** The only difference for the EVCC using its contract credential compared to the initial process (cf. Section 5.1.1) is that the respective private contract key is now stored within the TrustEV key hierarchy of the TPM 2.0, and, as the key access policy is defined, can only be loaded if this policy is fulfilled, similar to the first action in the Figure 5.6. Otherwise, this step is executed as usual.

### 5.4.3. Security Discussion

The security goal of the TrustEV architecture is to enable secure provisioning and storage of the PnC credentials on a potentially insecure EVCC and secure usage of these credentials during V2G service sessions, which achieve the required LoA [187]. In order to evaluate the security of the proposed architecture, we perform an informal security analysis regarding the requirements defined in Section 5.3.1.

The security of the TrustEV architecture strongly depends on the security of the HSM chosen as the trust anchor. Therefore, OEMs are expected to use only properly certified HSM products designed, reviewed and validated in a systematic way, equal to the Common Criteria EAL4+ procedures. TrustEV properties can only be achieved with security chips compliant to the TCG specifications for TPM 2.0, using CC EAL4+ certified hardware and qualified for automotive applications.

**Secure key storage.** In order to meet security requirement $RS_1$, private keys for OEM provisioning and contract credentials are stored on the EVCC within the TrustEV key hierarchy under protection of the vehicle's TPM. This protects the credentials against the car hacker, who wants to read out keys from the EVCC's storage or RAM. Moreover, the OEM provisioning key and the SRK used for key import are generated by the OEM in the vehicle's TPM. We already discussed that if an eMSP follows the provisioning strategy, where contract keys are generated in its backend, the eMSP is responsible to implement appropriate mechanisms to protect these keys from the backend hacker. Only for backward compatibility with eMSPs that do not support TrustEV, the keys may reside on the EVCC before being imported into the TPM. But even in this case, TrustEV strongly reduces the potential attack window for the car hacker.

**Secure cryptographic operations.** The fulfillment of the requirement $RS_2$ relies strongly on the security of the TPM 2.0 and TrustEV implementation on the EVCC, because PnC keys can be used only by the vehicle's TPM and only in its protected environment. The `TrustEV-EVCC` component can perform cryptographic operations like signing requests or authentication challenges only by calling the corresponding TPM commands.

**Key usage authorization.** The security requirement $RS_3$ is fulfilled by the TrustEV allowing the OEM to define arbitrary access policies including passwords for PnC keys. Before any key can be loaded, the TPM verifies the corresponding policy, and if this fails, the key cannot be used, e.g., for sending a request or charging authentication. The choice of the policy is up to the OEM. We opted for an exemplary authorization policy sealing the keys to the EVCC's firmware state. In this case, the TPM verifies, whether the EVCC is in a known and thus trustworthy state, by measuring firmware components during boot, before allowing access to the private OEM provisioning and contract keys. If any of the measured firmware components is compromised, the EVCC cannot use its PnC keys anymore. Runtime attacks, e.g., buffer overflows, where the keys are already loaded into memory and policies cannot be enforced anymore, still pose a security risk. It is worth noting that alternatively only the SRK can be sealed to a policy to restrict access to its children. Loading them would require the parent's policy to be fulfilled. Unfortunately, this opens a window for attacks using the `TPM2_EvictControl` command, where a TPM object is persistently stored in the Non-Volatile (NV) memory and can always be used without the need of loading it again and, thus, avoiding policy controls. For this reason, TrustEV requires the provisioning and contract keys to be directly sealed to a policy.

**Secure key provisioning.** In order to protect PnC credentials during provisioning and meet the requirement $RS_4$, TrustEV implements several protections. First, OEM provisioning keys are generated directly on the chosen EVCC's HSM during manufacturing and are stored within the protected TrustEV key hierarchy afterwards. Contract keys are generated and encrypted by eMSPs in such a way that only the EVCC's HSM can decrypt these keys after importing them into its protected environment. Moreover, to be able to request a new contract key, the EVCC needs to satisfy the conditions of a predefined policy controlling usage of the identity credentials, i.e., if the car hacker manipulated the EVCC, TrustEV will not allow the EVCC to send such request. Backward compatibility with eMSPs that do not support TrustEV reduces the assurance level of secure key provisioning, because the car hacker may access the unencrypted keys before they can be imported into the security module.

**Cryptographic agility.** As according to $RS_2$ all cryptographic operations are performed by the TPM, the fulfillment of requirement $RS_5$ depends on the capabilities of the deployed TPM 2.0 hardware. On the positive side, the TPM specification [114] defines the cryptographically agile interface and the TCG algorithm registry [116] includes most of the modern cryptographic algorithms. The algorithms required by ISO 15118-2 are widely supported, also, by the available hardware. Unfortunately, this is not the case with the ISO 15118-20, where the support of 521 bit keys and Elliptic Curve (EC) Curve448 are mandatory. The first requirement is not supported by TPM 2.0 chips on the market and the second requirement even is not part of the algorithm registry. In summary, TPM 2.0 provides cryptographic agility but it is currently limited with respect to the considered use case.

**Further considerations: protection against downgrade attacks.** Since the presented solution encodes the capability to use the TPM-based key import in a signed certificate (as the TrustEV certificate extension) even a man-in-the-middle adversary has no means to force the downgrade of the secured ISO 15118 protocol flow to the regular communication scheme without TrustEV.

### 5.4.4. TrustEV Prototype

In this section, we describe our prototypical implementation of the TrustEV architecture. The TrustEV PoC implements EVCC and SECC on a quad-core 1.2GHz 64bit platform with 1GB RAM running Linux kernel 4.14. The functionalities of eMSP/CPS are implemented on the same hardware as SECC for the sake of simplicity. The EVCC system is equipped with an Infineon OPTIGA™ SLM 9670, an automotive qualified and CC EAL4+ certified TPM 2.0. The PoC components communicate over PLC Stamp micro 2 EVBs; thus, powerline communication similar to a charging cable is supported. Our test-bed is shown in Figure 5.7.

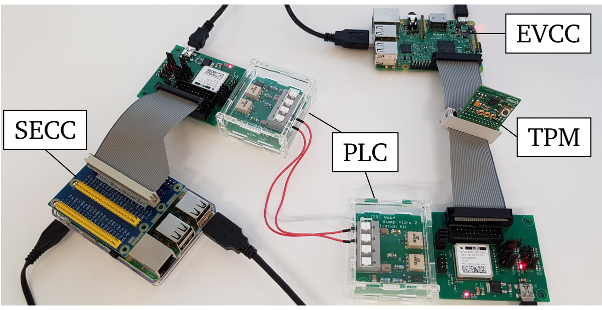


Figure 5.7: Credential Provisioning Test-bed Setup

For the execution of TPM commands, we use the TPM2-TSS[9] and initial certificates are generated with OpenSSL[10]. For the ISO 15118 communication between EVCC and SECC we use RISE V2G[11], which is extended with our TrustEV solution. The ISO 15118 messages are directly processed by the implemented eMSP/CPS functionalities on the SECC system. We implemented the TPM functionalities and certificate extension as described in Section 5.4.1 and the protocol steps for *EV Preparation*, *Generation of Contract Credentials*, *Installation of Contract Credentials*, and *Contract Credentials Usage* as described in Section 5.4.2.

The TrustEV operations during EV preparation are shown in Algorithm 2. First, the root key of the TrustEV hierarchy $\{SRK_{pub}; SRK_{pr}\}$ and under it the the OEM provisioning key $\{PC_{pub}; PC_{pr}\}$ are generated, whereby $PC_{pr}$ is sealed to an expected PCR value describing the EVCC's firmware state via *authPolicyPCR*. Next, the OEM provisioning certificate including $SRK_{pub}$ and the digest of *authPolicyPCR* in the TrustEV certificate extension is generated. The $SRK_{pr}$ is saved in the NV memory of the TPM, while the provisioning key is stored externally encrypted by the SRK.

```
/* OEM provisioning credential generation */
SRK = TPM2_CreatePrimary(SRK_ATTRIBUTES);
PC = TPM2_Create(SRK, PC_ATTRIBUTES, authPolicyPCR);
Cert_PC = generateCertificate(PC, SRK.pub, authPolicyPCR);
/* Save results */
TPM2_EvictControl(SRK);
save(PC, Cert_PC);
```

**Algorithm 2:** EV Preparation Steps

Algorithm 3 shows the operations for generating new contract credentials for direct TPM import performed by the

[9] https://github.com/tpm2-software/tpm2-tss
[10] https://github.com/openssl/openssl
[11] https://github.com/V2GClarity/RISE-V2G

eMSP that received a request with the TrustEV extension in the certificate.

```
/* Contract credential generation */
SRK_Pub, authPolicy = getByOID(certificate, TrustEV_EXT_OID);
CC = createContractKey();
CC_Pr_Enc, DHK_Pub = encryptCC(CC, SRK_Pub, authPolicy);
Cert_CC = generateCertificate(CC, SRK_Pub, authPolicy);
/* Save results */
save(CC_Pr_Enc, DHK_Pub, Cert_CC);
```

**Algorithm 3:** Generation of Contract Credentials

First, $SRK_{pub}$ and the policy digest are extracted from the certificate. Afterwards, a new contract keypair is generated and encrypted using the TPM's SRK from the TrustEV certificate extension, where the encryption method needs to be compliant with the `TPM2_Import` command using outer wrapper (cf. [114], Chapter 23.3). The encryption of the private contract key for TPM import is detailed in Algorithm 4. Finally, a new contract certificate is created, including $SRK_{pub}$ and the policy digest.

```
/* encryptCC(CC, SRK_Pub, authPolicy); */
CC_PublicArea = getPubArea(CC_ATTRIBUTES, CC.pub, authPolicy);
CC_Name = SHA256(CC_PublicArea);
CC_SensitiveArea = getSensitiveArea(CC.pr);
/* Generate keys */
DHK = createECDHephemeralKey();
Z = ECDH(DHK.pr, SRK_Pub);
seed = KDFe(SHA256, Z, "DUPLICATE", DHK.pub.X, SRK_Pub.X, 256);
/* Symmetric encryption key (aka session key) */
SK = KDFa(SHA256, seed, "STORAGE", CC_Name, NULL, 128);
/* HMAC key */
HK = KDFa(SHA256, seed, "INTEGRITY", NULL, NULL, 256);
/* Encrypt for direct import */
dupSensitive = AES_128_CFB(SK, 0, CC_SensitiveArea);
hmac = HmacSHA256(HK, dupSensitive||CC_Name);
CC_Pr_Enc = hmac||dupSensitive;
return (CC_Pr_Enc, DHK.pub);
```

**Algorithm 4:** TPM Import Encryption Function encryptCC()

In accordance with the Algorithm 4, first, a shared secret is generated using ECDH with $SRK_{pub}$ and a new ephemeral EC key. This shared secret is then provided as input to the predefined Key Derivation Function (KDF)s (cf. [114], Chapter 11.4.9) to generate the symmetric encryption key *SK* (aka ISO 15118 session key) and HMAC key *HK*. Next, the private contract key is encrypted using the SRK's symmetric algorithm AES in CFB mode for encryption and using the SRK's hash algorithm SHA256 to calculate an HMAC tag. This process includes the so called *name* of the contract key in the generation of the encryption key and the calculation of the HMAC. As per the TPM 2.0 specification [114], the *name* is a digest over the public area of the TPM key object, which in turn includes the attributes of the contract key, (cf. Section 5.4.1), the provided policy digest, as well as the public contract key itself. Thus, the contract key can only be imported into the EVCC's TPM if the EVCC provides the correct policy digest.

The new contract key structure enabled for direct import into TPM is presented in Figure 5.8. This private key structure contains an algorithm identifier (2 bytes), an authorization value padded (with zeros) to the length defined by the hash algorithm of the public contract key (32 bytes with SHA256 and ISO 15118-2; cf. [114] Chapter 27.7.3), a seed value (unused for asymmetric, non-storage keys; cf. [114] Table 27), the actual private key (32 bytes EC key with ISO 15118-2) and multiple size fields (all 2 bytes).

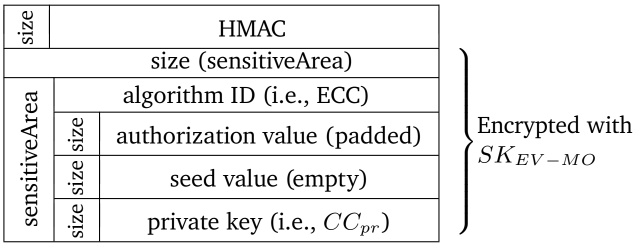


Figure 5.8: Encrypted Contract Key for Direct Import (adapted from [114], Figure 20)

The EV's process for installing a new contract key identified as a TPM key by the TrustEV extension is shown in Algorithm 5. The EV needs to load its storage key SRK, as well as the ECDH public key and the encrypted contract private key from the ISO 15118 response. Afterwards, it builds the public area for its new contract key from the predefined attributes, as well as the public key and policy digest included in the received certificate. The contract key is then imported into the TPM using the `TPM2_Import` command and saved externally encrypted by the storage key and therefore only accessible by the TPM.

```
/* Load data needed for import */
SRK = TPM2_TR_FromTPMPublic(SRK_Handle);
DHK_Pub = load(DHpublickey);
CC_Pr_Enc = load(ContractSignatureEncryptedPrivateKey);
CC_Pub = getPublicKey(Cert_CC);
/* Import key and save result */
CC = TPM2_Import(SRK, CC_Pub, CC_Pr_Enc, DHK_Pub);
save(CC);
```

**Algorithm 5:** Installation of Contract Credentials

To use the (private) contract key, an EV always has to provide the authorization policy. An example of this is shown in Algorithm 6, where the contract key is loaded to sign an ISO 15118 $AuthorizationReq$ message. In our prototype, we use the TPM2-TSS for loading the keys and for the creation of signatures. Instead of the PCR being extended by a measured boot, the contents of a file, representing the firmware, are hashed into the PCR used for sealing of the provisioning and contract keys. For the contract credential generation, we create and encrypt contract keys for direct import in Java on the SECC, since the SECC system also serves as the eMSP, and use Bouncy Castle[12] to generate the corresponding certificates. For the import, we use the TPM2-TSS. To integrate this solution into RISE-V2G on the EVCC, we call the compiled C programs using Java's ProcessBuilder and read the results from the generated files.

```
/* Load keys and sign AuthorizationReq */
SRK = TPM2_TR_FromTPMPublic(SRK_Handle);
CC = TPM2_Load(SRK, CC_Data);
authReqHash = SHA256(AuthorizationReq);
session = TPM2_PolicyPCR(PCR_SELECTION);
authReqSig = TPM2_Sign(CC, session, authReqHash);
```

**Algorithm 6:** Contract Credentials Usage

### 5.4.5. Evaluation

In the following, we evaluate TrustEV against the functional requirements from Section 5.3.2 in order to check if our solution meets its design goals. TrustEV addresses only the ISO 15118 compliant provisioning strategy and is not expected to fulfill requirements for the out-of-band provisioning.

**Minimal overhead.** In order to verify that TrustEV fulfills the functional requirement $RF_1$ and does not incur overheads that make it incompatible with the normal execution of ISO 15118, we analyze the increase in data/message sizes and execution times caused by our solution. We perform the measurements with our TrustEV prototype and compare the obtained results with the RISE-V2G implementation representing the ground truth. Table 5.1 summarizes our findings.

TrustEV's communication overhead comes from the new certificate extension in both OEM provisioning and contract certificates and larger encrypted private contract keys that need to be transferred between EVCC and SECC. As seen in Table 5.1, TrustEV adds about 14.3% of overhead to EVCC's certificate requests and about 4.8% to SECC's responses, which we consider acceptable.

Next, TrustEV introduces additional operations and, therefore, computational overhead in several steps of the provisioning process, i.e., generation, installation, and usage of contract credentials. In order to evaluate this overhead, we measure a waiting time and an authentication latency of the TrustEV prototype. We define *waiting time* as the time between sending a request for a new contract credential and having the newly generated contract credential successfully installed. The second parameter *Authentication latency* is the time between sending a challenge to authenticate the

[12] `https://www.bouncycastle.org/java.html`

Table 5.1: TrustEV Communication and Computational Overheads Summary

| Parameter | w/o TrustEV | w/ TrustEV | Overhead |
|---|---|---|---|
| Message size $CertInstallReq/CertUpdateReq$ | 812 bytes | 928 bytes | 116 bytes |
| Message size $CertInstallRes/CertUpdateRes$ | 3640 bytes | 3816 bytes | 176 bytes |
| Waiting time | 1233.8 ms | 1852.3 ms | 618.5 ms |
| Authentication latency | 58.1 ms | 647.3 ms | 589.2 ms |

Table 5.2: Conformance to XSD Constraints in ISO 15118-2

| Parameter | w/o TrustEV | w/ TrustEV SHA-1/SHA256 | Conform |
|---|---|---|---|
| Certificate size | $\leq$800 bytes | 590 bytes | + |
| Encrypted private key size $privateKeyType$ | 48 bytes | 84/108 bytes | − |
| +Algorithm identifier | absent | 2 bytes | |
| +Authorization value | absent | 20/32 bytes | |
| +Private key | 32 bytes | 32 bytes | |
| +Size fields | absent | 2 bytes $*$ 5 | |
| +HMAC | absent | 20/32 bytes | |

EVCC and receiving the response containing the challenge signed by the EVCC with the correct contract key. We use the function $System.nanoTime()$ in Java to perform measurements of the waiting time and the authentication latency. Table 5.1 shows the average values for 100 measurements. Though the overhead of about 50.1% with regard to the waiting time is significant, it is still below the critical 5 seconds timeout for ISO 15118's installation response and is not perceptible to PnC users. However, a part of this overhead results from the relatively slow TPM import operation executed after receiving the response and thus not affecting the timeout directly. Similarly, the much increased authentication latency, due to loading the TPM keys and TPM-assisted signing, is still below 1 second and is not expected to cause user inconvenience. Even with this overhead, our prototype does not exceed ISO 15118's EVCC performance and SECC sequence timeouts equal to 40 seconds and 60 seconds, respectively.

**Conformance to the ISO 15118 protocol specifications.** In order to verify the fulfillment of $RF_2$, we first analyze, which requirements of the ISO 15118 specification concern TrustEV. As discussed above, additional security functionality results in the increased operation latency and sizes of the protocol messages related to credential provisioning. ISO 15118 specifies strict time and data size constraints, which when exceeded lead to SECC terminating the PnC service session. In order to be conform to ISO 15118-2[13] the TrustEV prototype needs to meet (i) timing requirements (cf. [38], Table 109), (ii) constraints on the sizes of XML data structures (cf. [38], Annex C.6), and (iii) the specified message sequencing for EVCC and SECC (cf. [38], Section 8.8.4).

According to the performance measurements reported in Table 5.1, our prototype meets the relevant timing requirements, and no further analysis is needed. Similarly, the above analysis of message sizes for $CertInstallReq/$ $CertUpdateReq$ and $CertInstallRes/$ $CertUpdateRes$ already shows that TrustEV changes the respective XML structures. We list the relevant constraints and the results of our measurements in Table 5.2. The plus sign (+) in the last column indicates that the TrustEV prototype meets the constraint, while the minus sign (−) stays for the opposite.

Though adding the TrustEV extension to PnC certificates does not violate the message size limitations in our measurements, this result might not work for those organizations that already use up all available certificate size with their specific information. However, our prototype proves that a standard-conform implementation is possible.

As for the XML data structure for the encrypted private key, Table 5.2 reveals that TrustEV does not meet the respective constraint. The reason is that contract keys encrypted for a direct TPM import must follow a predefined format, a TPM

[13] Though TrustEV also aims to be conform to ISO 15118-20, we do not perform the corresponding measurements, because our "ground truth" implementation RISE V2G does not support this new edition at the moment of writing. Instead, we report on the integration of the TrustEV architecture into the final draft (FDIS) in Section 5.4.6.

private key structure (cf. Figure 5.8), that adds several bytes of prefix to the key before it is encrypted and an HMAC to the resulting ciphertext. This gives an encrypted TrustEV contract key with the minimum size of 84 bytes when the hash algorithm SHA-1 is used and 108 bytes with SHA256, in contrast to the expected maximum size of 48 bytes specified in the XML schema (XSD) of ISO 15118 [38]. Moreover, the TPM 2.0 specification requires AES-128 in CFB mode instead of CBC mode used in ISO 15118-2 (cf. [38], [V2G2-815]) or GCM mode used in ISO 15118-20 (see [183, 39]). This puts an extra burden of supporting an additional encryption format and cipher mode on the eMSP's backend system and could lead to incompatibilities if intermediate actors between eMSP and EVCC, i.e., CPS or SECC, validate the size of the encrypted key before they forward it. Notably, our reference implementation RISE V2G does not encounter any problems with this constraint's violation. Thus, whether the communication fails or not, is implementation-dependent. ISO 15118 generally allows de-/encoding of undefined XML elements/attributes in EXI (cf. [38], [V2G2-101]) and does not define any relevant validation procedures, e.g., for SECC or error cases for this, either. In the backend, OCPP 2.0 simply forwards ISO 15118 response messages as Base64 encoded binary data without further processing.

TrustEV does not change any ISO 15118 message sequences compared to the specification [38, 183, 39]. Thus, apart from the issue with the XML data size for encrypted private contract keys, this requirement is mostly fulfilled.

**Backward compatibility.** A very important aspect of ISO 15118 is that technological decisions are open to the implementers, unless they are critical for safety, interoperability, or security and thus need to be specified unambiguously. Our TrustEV security extension is supposed to be deployed in parallel to the existing ISO 15118 compliant products and should allow eMSPs to render services to any clients. In the following, we check if TrustEV satisfies the requirement $RF_3$ and can be integrated into the emerging V2G infrastructure without causing service failures.

TrustEV uses the certificate extension in the OEM provisioning and contract certificates to reliably notify the eMSP that the vehicle is equipped with a highly secure HSM and wishes the TrustEV protocols to be used for provisioning of new credentials. It is up to the eMSP to decide if it supports this feature. However, if the vehicle's OEM decides to invest into a higher security standard, we expect the eMSP to second this decision, considering that it is more costly for EVCC to support multiple encryption formats and modes. But if the eMSP does not recognize this extension, it can proceed as usual with the public OEM provisioning or contract key from the certificate request, because the TrustEV extension is marked as non-critical. In this case, the eMSP also does not include the TrustEV extension into the newly issued contract certificate to notify the EVCC that the standard provisioning procedure is executed. The EVCC can then follow the installation steps according to ISO 15118 without TrustEV. We have learned from OEMs that it is uncommon to decrypt and store sensitive cryptographic keys unprotected, thus, the EVCC equipped with TPM 2.0 is likely to reject non-TrustEV keys. If the EVCC does not support TrustEV, its OEM provisioning certificate does not contain the TrustEV extension. In this case, the eMSP does not have other choice but to follow the standard provisioning process. To conclude, we consider this requirement fulfilled, even though the eMSPs may be forced to support TrustEV by their clients.

### 5.4.6. TrustEV Integration into the Upcoming ISO/FDIS 15118-20

We presented our TrustEV security architecture to the members of the Security Task Force organized within ISO/TC 22/SC 31/JWG 1 "Joint ISO/TC 22/SC 31 – IEC/TC 69 WG: Vehicle to grid communication interface (V2G CI)" responsible for the development of the ISO 15118 standard series. Our proposal for change describing the TrustEV extensions to the protocol successfully passed the committee voting and the TrustEV solution has been officially integrated into the upcoming edition ISO/FDIS 15118-20 (see [39], Chapter 7.9.2.5) with the expected release date in Q2 2022.

Since this new edition requires 256-bit security and different ECC algorithms – secp521r1 as a default curve and Curve448 as an alternative curve – we had to update the TPM key profiles for the TrustEV key hierarchy as shown in Table 5.3. As we already mentioned during the security discussion in Chapter 5.4.3, the TCG algorithm registry [116] does not support Curve448 and, thus, our TrustEV solution can only be used with the default EC if the respective hardware is available. Since this feature is optional, this partial compatibility of the TrustEV protocol extension to the specification reduces the acceptance among OEMs. The decision to invest into a better security for credential provisioning on the OEM side is backed up by the standard, which demands from eMSPs to encrypt the private contract key for the TPM import if the TrustEV extension is present in the OEM provisioning certificate provided by the EVCC (see [39], [V2G20-2595]).

The maximum certificate size has been extended to 1.600 bytes to allow for the TrustEV extension. The updated OEM provisioning certificate profile is shown in Figure 5.9.

Table 5.3: Public Area Attributes of the TPM 2.0 Keys in ISO/FDIS 15118-20

| Attribute | Storage Root Key | OEM Provisioning Key | Contract Key |
|---|---|---|---|
| type: | TPM2_ALG_ECC | TPM2_ALG_ECC | TPM2_ALG_ECC |
| nameAlg: | TPM2_ALG_SHA512 | TPM2_ALG_SHA512 | TPM2_ALG_SHA512 |
| objectAttributes: | fixedTPM, fixedParent,<br>restricted, decrypt,<br>sensitiveDataOrigin,<br>userWithAuth | fixedTPM, fixedParent,<br>sign, decrypt,<br>sensitiveDataOrigin,<br>adminWithPolicy | sign,<br>sensitiveDataOrigin,<br>adminWithPolicy |
| authPolicy: | n/a | OEM defined | OEM defined |
| curveID: | TPM2_ECC_NIST_P521 | TPM2_ECC_NIST_P521 | TPM2_ECC_NIST_P521 |

<table>
<tr><td colspan="5">OEM Provisioning Certificate ($Cert_{PC}$) in ISO/FDIS 15118-20</td></tr>
<tr><td colspan="2">Version:</td><td colspan="3">X.509v3 (0x2)</td></tr>
<tr><td colspan="2">Serial Number:</td><td colspan="3">12345 (0x3039)</td></tr>
<tr><td colspan="2">Signature Algorithm:</td><td colspan="3">ecdsa-with-SHA512</td></tr>
<tr><td colspan="2">Issuer:</td><td colspan="3">CN=OEMSubCA2, O=ISO</td></tr>
<tr><td rowspan="2">Validity</td><td>Not Before:</td><td colspan="3">Aug 20 08:40:32 2021 GMT</td></tr>
<tr><td>Not After:</td><td colspan="3">Aug 20 08:40:32 2051 GMT</td></tr>
<tr><td colspan="2">Subject:</td><td colspan="3">CN=8AAA2B3C4D5E6F7G92, O=ISO</td></tr>
<tr><td rowspan="3">Subject Public Key Info</td><td>Public Key:</td><td colspan="3">OCTET STRING ($PC_{pub}$)</td></tr>
<tr><td>Algorithm:</td><td colspan="3">id-ecPublicKey</td></tr>
<tr><td>Parameters:</td><td colspan="3">namedCurve secp521r1</td></tr>
<tr><td rowspan="10">X509v3 Extensions</td><td>Basic Constraints:$^{c}$</td><td colspan="3">CA:FALSE</td></tr>
<tr><td>Key Usage:$^{c}$</td><td colspan="3">Digital Signature, Key Agreement</td></tr>
<tr><td>Subject Key Identifier:$^{c}$</td><td colspan="3">64 bit ID of $PC_{pub}$</td></tr>
<tr><td>Authority Key Identifier:$^{c}$</td><td colspan="3">64 bit ID of issuer's public key</td></tr>
<tr><td>CRLDistributionPoints:$^{c}$</td><td colspan="3">URI:http://example.com/example.crl</td></tr>
<tr><td>AuthorityInfoAccess (OCSP):$^{c}$</td><td colspan="3">URI:http://ocsp.example.com/</td></tr>
<tr><td rowspan="4"><b>Subject Information Access:</b>$^{nc}$</td><td rowspan="2">$SRK_{pub}$</td><td colspan="2"><b>OID:1.0.20.0.4</b></td></tr>
<tr><td colspan="2"><b>TPM 2.0 EC Storage Root Key 1056 bit OCTET STRING</b></td></tr>
<tr><td rowspan="2">$Pol$</td><td colspan="2"><b>OID:1.0.20.0.5</b></td></tr>
<tr><td colspan="2"><b>TPM 2.0 SHA512 Policy Digest 512 bit OCTET STRING</b></td></tr>
<tr><td rowspan="2">Signature</td><td>Algorithm:</td><td colspan="3">ecdsa-with-SHA512</td></tr>
<tr><td>Value:</td><td colspan="3">OCTET STRING</td></tr>
</table>

Figure 5.9: Provisioning Certificate with Custom Extension in ISO/FDIS 15118-20

The TrustEV certificate extensions were assigned official OIDs as shown in Algorithm 7.

```
/* Public Storage Key of EVCC's TPM */
id-tpmStorageKey OBJECT IDENTIFIER = id-15118-20-ad tpmStorageKey(4);
/* Policy Digest for EVCC's TPM Contract Keys */
id-tpmPolicyDigest OBJECT IDENTIFIER = id-15118-20-ad tpmPolicyDigest(5);
```

**Algorithm 7:** TrustEV Extensions OIDs

In order to transfer the contract key encrypted for the TPM import, we introduced a new XML type *tpm_EncryptedPrivateKeyType* into the schema (see Listing 5.1) to preserve the strongly typed message format used in ISO 15118. An element of this type can be added by the eMSP to the *SignedInstallationData* structure containing contract credential data package and sent via the $CertInstallRes$ message.

Listing 5.1: Encrypted TPM Key Type

```
<xs:simpleType name="tpm_EncryptedPrivateKeyType">
  <xs:restriction base="xs:base64Binary">
    <xs:length value="206"/>
  </xs:restriction>
```

```
</xs:simpleType>
<xs:complexType name="SignedInstallationDataType">
  <xs:sequence>
    <xs:element name="ContractCertificateChain" type="ContractCertificateChainType"/>
    <xs:element name="DHPublicKey" type="dhPublicKeyType"/>
    <xs:choice>
      <xs:element name="SECP521_EncryptedPrivateKey" type="secp521_EncryptedPrivateKeyType"/>
      <xs:element name="X448_EncryptedPrivateKey" type="x448_EncryptedPrivateKeyType"/>
      <xs:element name="TPM_EncryptedPrivateKey" type="tpm_EncryptedPrivateKeyType"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Id" type="xs:ID" use="required"/>
</xs:complexType>
<xs:element name="SignedInstallationData" type="SignedInstallationDataType"/>
```

As soon as a hardware TPM with the necessary cryptographic capabilities is available, we plan to evaluate TrustEV with the new parameters required for the upcoming edition.

### 5.4.7. TrustEV Summary

In this section, we introduced TrustEV, an HSM-based security architecture for secure provisioning and protecting PnC credentials stored in the vehicle. Our solution addresses the standard ISO 15118 provisioning approach, where contract credentials are generated and delivered by eMSPs on request received over V2G infrastructure from the registered EVs of their customers. Thereby, TrustEV should remain conform to the ISO 15118 specifications. We defined the TrustEV components for an HSM-equipped EVCC and the backend systems of its OEM and eMSP. In order to enable the authenticated delivery of the required security parameters from the OEM to the eMSP, OEM provisioning and contract certificates received an optional TrustEV extension carrying an alternative encryption key and key usage policy. We also described the extensions to the ISO 15118 protocol necessary to seamlessly integrate TrustEV into the protocol flow. Within the vehicle, the EVCC is equipped with a TPM 2.0 and runs the TrustEV component, which processes the extension fields and invokes TPM functions for secure key storage, key import, and cryptographic operations. Since TPM 2.0 and the corresponding software stack are specified as open standard and both CC EAL4+ certified hardware and open-source implementations are available, its usage to implement HSM functionalities in TrustEV ensures that our evaluation results are reproducible and can be applied for benchmarking automotive HSMs being developed for the same purpose. With TrustEV, private keys of PnC credentials are protected against car hackers who have an unrestricted access to an EV, because OEM provisioning keys never leave the protected environment of the vehicle's TPM and contract keys are end-to-end encrypted for the target TPM and cannot be decrypted otherwise. In addition, TrustEV provides a mechanism to protect key usage when the EVCC is compromised, by sealing private keys to an OEM defined policy, e.g., to a firmware boot state. If a V2G actor does not support TrustEV, this extension can be ignored and the legacy approach simply be used instead. Our implementation and evaluation of TrustEV with a hardware TPM 2.0 proved its security and feasibility. The introduced overhead was, with one exception, within the limits of the ISO 15118 specification. Only the size of the encrypted contract key for the direct TPM import exceeded the specified value, because of the more secure import format. This may lead to incompatibilities if intermediate actors validate the size of the encrypted key or the overall message before forwarding it.

The advantages of the TrustEV solution for secure provisioning of PnC credentials convinced the standardization committee ISO/TC 22/SC 31/JWG 1 responsible for the development of the ISO 15118 standard series and our proposed security extension will be officially supported in the upcoming edition of ISO/FDIS 15118-20 [39]. However, the backend hacker is only partially addressed by TrustEV, i.e., only for OEM backend systems. In the next section we consider the attacks on eMSP backend systems that generate and at least temporary store all private contract keys of all customers. We investigate an approach, where contract keys are securely generated within the vehicle's HSM and used to request only the respective contract certificate from an eMSP.

## 5.5. HIP: HSM-based Identities for Plug-and-Charge

In this section, we introduce our new security architecture HIP for secure provisioning of PnC credentials using the alternative provisioning strategy, where the keypair of the vehicle's contract credential(s) is generated and stored in the EVCC, and eMSP creates and delivers a new x.509v3 certificate for the public contract key included in the request from its customer's EV sent during an active PnC session. HIP aims to protect PnC credentials both from the car hacker and backend hacker (cf. Section 5.2), thus, reducing residual security risks for eMSPs. As only public keys need to be stored in the backend systems of the OEM and eMSP, even a successful attack on these systems cannot compromise PnC credentials. HIP builds on the TrustEV security architecture discussed in the previous section and similarly relies on a certified HSM for *secure generation* and storage of cryptographic keys, and cryptographic operations performed on the vehicle's EVCC to meet the security requirements from Section 5.3.1 and achieve the required LoA. By generating cryptographic key in an HSM so that these keys never leave this HSM, HIP also realizes the respective NIST recommendations [188].

When outsourcing key generation to the vehicles of its customers, an eMSP needs an assurance that contract credential keys are generated and stored in an HSM with the expected security guarantees. Beside the protection from the car hacker, for example, the conformance to the Payment Card Industry Data Security Standard (PCI DSS) may be mandatory, if contract credentials are qualified as payment credentials. For this purpose, HIP provides the capability to securely bind contract keys to the HSM, where they were generated, in such a way that the EVCC cannot use the corresponding contract certificate, e.g., for service authentication, unless the private contract key was generated and is stored in this HSM. Additionally, our solution can guarantee that contract keys were generated in the same HSM as the OEM provisioning key and with expected attributes and access policy. The binding between the HSM, the EVCC and the EVCC's provisioning credentials needs to be certified by the vehicle's OEM. Thus, if the eMSP trusts the OEM provisioning credentials (e.g., based on the OEM's certificate policy), the eMSP can also trust that contract keys are adequately secured. The novelty of the HIP architecture is in enabling secure contract key generation in the vehicle, nevertheless, it adopts several features of TrustEV introduced in the previous section. HIP uses the TrustEV key hierarchy for generation and storage of PnC keys; certificate extension to prove the binding between the HSM, the EVCC, and the OEM provisioning credentials to an eMSP; and key access policies. Driven by the same motivation, to develop a solution ready for international standardization and provide reproducible benchmarks, we adapt the openly specified and CC certified TPM 2.0 to define HSM API and credential protection protocol for HIP, too.

As the provisioning strategy supported by HIP does not comply with ISO 15118, the main challenge with regard to functional requirements from Section 5.3.2 is to ensure that the introduced changes do not disturb V2G communication and are backward compatible. In order to enable the proposed security functionality, changes are required in the initial preparation of the EV performed by the OEM, in the generation of the contract keypair, during creation and provisioning of contract certificates for EV enrolling with own contract keys, OEM provisioning and contract certificate profiles, and the charging process to make use of the TPM. As ISO 15118 does not allow for the transfer of EVCC's public keys except for its OEM provisioning key, HIP also alters ISO 15118 message format. Thereby, the message sequencing remains unchanged. In order to verify that this new extension does not affect the core functionality of ISO 15118, we evaluate its security and performance using an upgraded TrustEV proof-of-concept implementation and test-bed setup.

In summary, we make the following main contributions:

- We present HIP, a new security architecture design providing beside secure storage and usage of PnC credentials achieved with TrustEV also *secure and verifiable generation of all PnC cryptographic keys* in an EV equipped with a HSM and *trustworthy credential enrollment* at an eMSP, whereby the eMSP receives an assurance that the customer's vehicle can use the contract credential only if the corresponding keys are generated with expected properties and are held in the same HSM as the OEM provisioning key used to sign the certificate request.
- We provide a concept for integrating the TPM 2.0 credential protection feature during secure provisioning of contract certificates compliant and backward compatible with ISO 15118-2 and ISO 15118-20.
- We analyze and discuss the security properties of HIP considering strong adversaries with control over the EVCC, communication channels, and eMSP backend systems.
- We evaluate several performance aspects of HIP and discuss the feasibility of the solution under ISO 15118 constraints.

This section is organized as follows: In Section 5.5.1, we introduce the components of our new security architecture HIP and in Section 5.5.2 describe the extensions to ISO 15118 necessary for the integration of the proposed security

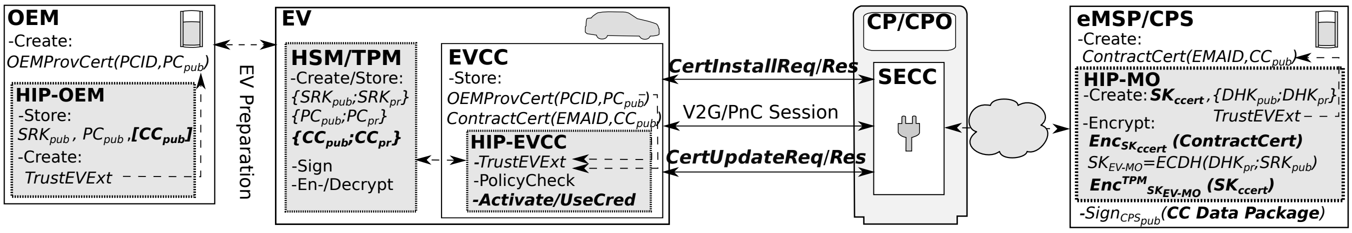


Figure 5.10: HIP Security Architecture

functionality. In Section 5.5.3, an informal security analysis of the proposed architecture is performed. Section 5.5.4 provides considerations regarding key revocation in ISO 15118 and the specifics of HIP in this regard. We describe our prototypical implementation and performance evaluation in Section 5.5.5. Finally, Section 5.5.6 provides a summary of the achieved results.

### 5.5.1. HIP Components

The goal of the proposed security architecture HIP is enable the EVCC's enrollment at an eMSP with keys generated securely in its HSM so that the private key never leaves the protected environment and does not need to be stored in the intermediate systems. Moreover, as in ISO 15118 theoretically any party can send an encrypted "blob" to the EVCC, installing an encrypted contract poses a potential threat to the EVCC. Whereas in regular ISO 15118 an eMSP generates contract keys and the corresponding certificate and sends them to the EVCC, with HIP, the EVCC generates the keypair of the contract credential HSM and afterwards requests the eMSP to issue a X.509v3 certificate for the public contract key. HIP is based on the previously introduced TrustEV architecture (cf. Section 5.4) as Figure 5.10 depicting components of HIP shows. Similar to TrustEV, the OEM, the EVCC, and the eMSP need to implement the additional security features to enable HIP. The newly added functionality extending TrustEV is shown in "bold italic" in Figure 5.10, where square brackets mark optional functions.

The `HIP-OEM` component inherits the generation of the SRK-based key hierarchy in the vehicle's HSM during EV manufacturing. The keys created by the OEM are the asymmetric parent key $\{SRK_{pub}; SRK_{pr}\}$ and the OEM provisioning key $\{PC_{pub}; PC_{pr}\}$. Now, the contract key $\{CC_{pub}; CC_{pr}\}$ can optionally be created during manufacturing and the public key kept in the OEM backend, too. HIP also uses the TrustEV extension `TrustEVExt` to realize the binding between the vehicle's HSM, i.e., its SRK, the OEM provisioning key and the key access policy. The `HIP-EVCC` component beside providing the interface to the HSM, is now additionally responsible for generating the contract keypair in this HSM and sending $CC_{pub}$ to the eMSP within the modified certificate request message. As the response does not include the encrypted private key anymore, it also needs to be processed by this component separately. The `HIP-MO` component in the eMSP backend uses both the custom certificate extension and certificate request message containing public key to determine that the EVCC supports HIP and wishes to enroll with own contract key. If the eMSP decides to meet the request and allow the EVCC to use the self-generated key, it will issue the contract certificate for this key and if a version of ISO 15118-2 is executed, add the custom certificate extension `TrustEVExt` to this certificate for future updates. Next new function of `HIP-MO` is the indirect proof of possession of the private contract key by the EVCC. This function is based on the credential protection protocol of TPM 2.0 [114]. The eMSP encrypts the newly generated certificate using a newly generated symmetric key, and then encrypts this symmetric key in such a way that only EVCC having access to the HSM storing both the OEM provisioning key and the contract key is able to decrypt the resulting ciphertext. Compared to ISO 15118, the eMSP now needs to implement an additional encryption method for the symmetric key and perform two cryptographic operations. Compared to TrustEV, the component needs to parse a custom certificate request, create a certificate for the provided key, and encrypt this certificate and the corresponding encryption key. Obviously, the format of the response message also needs to be customized to include these data. In order to define the HSM interface, we use the TPM 2.0 [114] for secure key generation, storage and usage. In order to restrict the access to the keys, we include the enhanced authorization functionality of the TPM, i.e., access policies, and the credential protection feature already mentioned above.

In summary, with minimal changes to ISO 15118, the HIP security architecture allows an EVCC to request certificates for its locally generated contract keys. Thereby, the eMSP is provided with a posteriori proof-of-possession and an assurance that these keys are created in the TPM holding the OEM provisioning key of the EVCC, with the expected attributes, and sealed to the authentic policy. In the following, we describe HIP extensions in detail.

**HIP Key Hierarchy and Profiles.** For secure generation and storage of cryptographic keys with HIP, we adopt the TrustEV key hierarchy described in Section 5.4.1, where Storage Root Key is serving as the parent for OEM provisioning and contract keys. In contrast to the TrustEV architecture, HIP allows all keys used for PnC to be generated inside the EVCC's TPM and to be optionally protected by a *policy* defined by the vehicle's OEM and enforced by the TPM. Table 5.4 shows the TPM key profiles for the PnC keys managed by HIP, which includes their TPM-specific parameters and attributes: the OEM provisioning keypair $\{PC_{pub}, PC_{pr}\}$, the contract credential keypair $\{CC_{pub}, CC_{pr}\}$, and asymmetric SRK $\{SRK_{pub}, SRK_{pr}\}$. The cryptographic algorithms listed in the table comply with the ISO 15118 specification [38], which mandates the usage of ECC keys and the secp256r1 (aka NIST_P256) curve. Notably, HIP is only limited by the registry of algorithms supported by the TPM [116] and can also be used with the secp521r1 curve required by the upcoming edition ISO 15118-20 [39].

Table 5.4: Public Area Attributes of the HIP Keys for ISO 15118-2

| Attribute | $SRK$ | $PC$ and $CC$ |
|---|---|---|
| type: | TPM2_ALG_ECC | TPM2_ALG_ECC |
| nameAlg: | TPM2_ALG_SHA1 | TPM2_ALG_SHA256 |
| objectAttributes: | fixedTPM, fixedParent, restricted, decrypt, sensitiveDataOrigin, userWithAuth, noDA | fixedTPM, fixedParent, sign, decrypt, sensitiveDataOrigin |
| authPolicy: | n/a | TPM2_PolicyAuthorize |
| curveID: | TPM2_ECC_NIST_P256 | TPM2_ECC_NIST_P256 |

The profile of the storage key SRK follows the TCG's recommendation for a default storage primary key template [189], with the exception of using SHA-1 as *nameAlg*. As we show later, this adjustment is required only to comply with the maximum size of encrypted contract keys in the ISO 15118 contract installation/update response messages. Otherwise, SHA256 can be used as *nameAlg* for the SRK, similar to the PnC keys. Since SHA-1 is only used as basis of the Key Derivation Function (KDF) and HMAC calculation (cf. Section 5.5.2), where collision resistance is not required, its applications is still considered secure [188].

The $PC$ and $CC$ keys carry the attributes *fixedTPM* and *fixedParent* to forbid the private key export and shifts within the key hierarchy (cf. [114], Section 25) and *sensitiveDataOrigin* to denote that they were generated by the TPM and not imported. The attributes *sign* and *decrypt* denote the usage type for these keys according to ISO 15118. In order to allow the OEM to change a key's policy after its generation, HIP uses so-called *PolicyAuthorize* as a default policy for the keys [113]. This TPM policy is basically a placeholder for any key usage policy approved by the OEM, and, thus, provides a flexible mechanism of updating usage conditions associated with a particular key. For example, the OEM can seal the contract key to the policy *PolicyPCR* enabling access to the keys only if the EVCC's firmware booted into an expected state (cf. [114], Section 19.7 Enhanced Authorization). Generally speaking, the eMSP cannot validate, which policy is associated with the contract keys generated by the OEM, because it only knows its hash. With *PolicyAuthorize*, the OEM can adjust this policy at will. In contrast to the $SRK$, the restricted attribute is not set since $PC$ and $CC$ are general-purpose keys and not storage keys.

**Contract Certificate Request.** Since the ISO 15118's request message for contract certificate installation $CertInstallReq$ or update $CertUpdateReq$ does not include such parameter as public contract key necessary for creating the respective certificate by the eMSP, this parameter needs to be added to the specified format. Therefore, HIP extends the format of these messages with the optional field *PublicContractKey*, encoded as a 64-octet Base64 string. Figure 5.11 illustrates the change in the $CertInstallReq$ message in bold.

To ensure that the eMSP processes this custom message correctly, the usage of HIP may be signalized by the EVCC via a dedicated minor protocol version number for this protocol extension, which can be negotiated during PnC session establishment. According to ISO 15118-2 [38], the SECC shall except such connection from an EVCC, if the major protocol version matches. In this case, the SECC shall also forward additional or unknown data elements sent by the EVCC to SAs (cf. [38], Section 8.2.1).

**Custom Certificate Extension.** In HIP, we use the TrustEV certificate extension (cf. Section 5.4.1) as a custom extension in OEM provisioning certificate and, in the case of ISO 15118-2, in contract certificate to provide an authentic binding

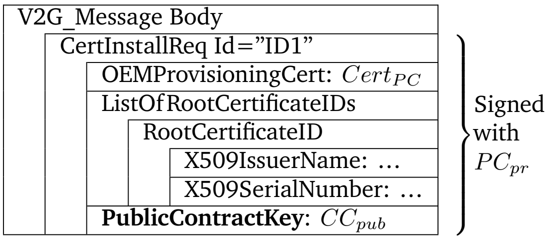


Figure 5.11: Extension of the ISO 15118 $CertInstallReq$ Message

between the information about vehicle's TPM (via SRK), OEM provisioning key residing in this TPM, and OEM defined policy for key access. We assume that an eMSP can verify this information, e.g., using the official certificate policy of the OEM. Since in ISO 15118, the EVCC authenticates its certificate request either with the OEM provisioning key or previously enrolled contract key, the eMSP cannot infer from this request, if the EVCC/TPM holds the private contract key corresponding to the public contract key received with the request. Thus, HIP adopts the credential protection mechanism of TPM 2.0 that provides the eMSP with an indirect proof-of-possession, because the newly issued contract certificate can only be used if the EVCC indeed fulfills this condition. For this mechanism to work, the eMSP requires some additional information about the EVCC's TPM such as the $SRK_{pub}$ and the digest of the access policy $Pol$ of the $PC$ included in the custom (TrustEV) certificate extension of the identity credential provided by the EVCC. The $SRK_{pub}$ is used as decryption target and the policy digest is used as part of the credential datatype encryption. This approach ensures that keys for the contract certificates are created in the EVCC's TPM with defined attributes and bound to a policy defined by the OEM.

### 5.5.2. HIP Protocol Extensions

For using HIP within the ISO 15118 credential provisioning process (cf. Section 5.1.1), we define extensions to the original protocol, where the new security functionality is engaged. Below, we follow the previously discussed standard steps and describe necessary changes.

**EV Preparation.** This step is carried out very much alike it is described for the TrustEV solution (cf. Section 5.4.2). During manufacturing, the OEM uses the EVCC's TPM to create the protected storage hierarchy, including the SRK as parent for other PnC keys according to their profiles described in Section 5.5.1. During generation of the OEM provisioning key $PC$, the OEM includes the respective access policy $Pol$ for this key (and also contract keys) using the method *TPM2_PolicyAuthorize* (cf. Section 5.5.1 and [114], Section 19.7.11). As discussed before, it allows the OEM to change the policy afterwards. The public keys $SRK_{pub}$ and $PC_{pub}$ are read out by the manufacturer in a secure environment to generate the OEM provisioning certificate $Cert_{PC}$ including the custom extension with additional security parameters $SRK_{pub}$ and digest of the policy $Pol$ as described in Section 5.5.1.

**Generation of Contract Credentials.** With HIP, the generation of contract credentials is distributed between the EVCC and the eMSP. First, the HIP-EVCC component of the EVCC uses the TPM to generate a new keypair $\{CC_{pub}, CC_{pr}\}$ underneath the SRK and seal it to the policy $Pol$. The public key is included in the certificate installation or update request and sent to the eMSP of the EV user as described below.

In contrast to ISO 15118, the eMSP only needs to issue an X.509v3 certificate $Cert_{CC}$ for the public contract key $CC_{pub}$ provided by the EVCC as part of the custom certificate installation or update request. This operation is performed by the provider's CA in the usual way (cf. GenCert in Figure 5.13), except for ISO 15118-2, where the custom extension containing $SRK_{pub}$ and the policy digest is added to the contract certificate for future credential updates (cf. [38], Section 8.4.3.10). Next, the eMSP initiates the credential protection protocol (cf. [114], Section 24) for the newly generated certificate. The corresponding data structures and cryptographic functions are detailed in Figure 5.12, while the sequence diagram of the protocol is shown in Figure 5.13. To enable credential protection, the contract certificate is encrypted using a newly generated symmetric key $SR_{ccert}$ and the algorithm AES-CBC-128 for ISO 15118-2 and AES-GCM-256 for ISO 15118-20 (cf. EncCert in Figure 5.13), with the initialization vector IV randomly generated before encryption. The contract certificate serves in terms of the TPM credential protection as "challenge", the encryption key thus needs to be wrapped with the public key of the TPM, i.e., SRK. The key wrapping is done as follows: the symmetric

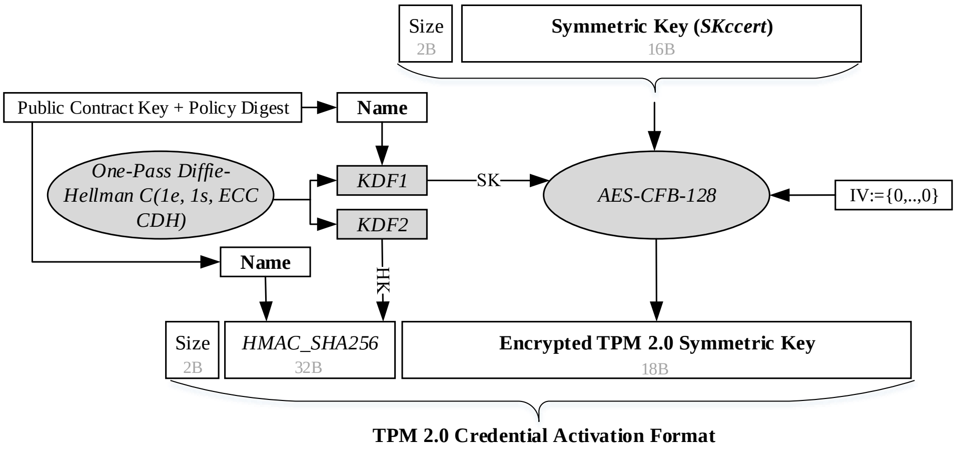

Size
2B
Symmetric Key (SKccert)
16B
Public Contract Key + Policy Digest
Name
One-Pass Diffie-Hellman C(1e, 1s, ECC CDH)
KDF1
SK
AES-CFB-128
IV:={0,..,0}
KDF2
Name
HK
Size
2B
HMAC_SHA256
32B
Encrypted TPM 2.0 Symmetric Key
18B
TPM 2.0 Credential Activation Format

Figure 5.12: Encryption of Symmetric Key for Credential Activation for ISO 15118-2 (adapted from [114], Figure 26)

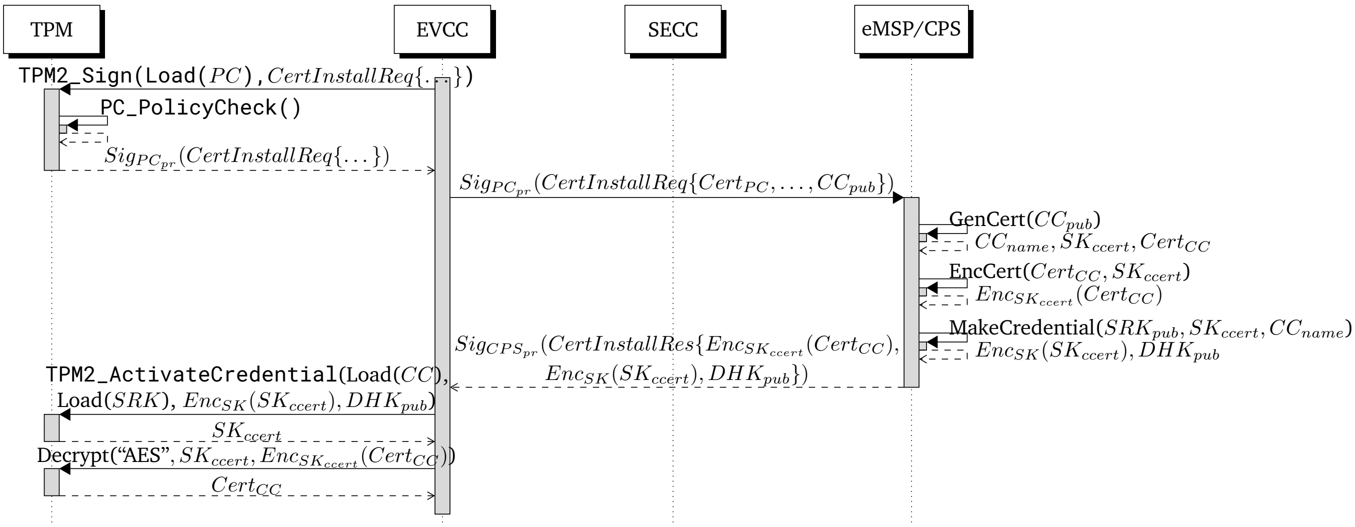

TPM
EVCC
SECC
eMSP/CPS
TPM2_Sign(Load($PC$), $CertInstallReq\{\ldots\}$)
PC_PolicyCheck()
$Sig_{PC_{pr}}(CertInstallReq\{\ldots\})$
$Sig_{PC_{pr}}(CertInstallReq\{Cert_{PC},\ldots,CC_{pub}\})$
GenCert($CC_{pub}$)
$CC_{name}, SK_{ccert}, Cert_{CC}$
EncCert($Cert_{CC}, SK_{ccert}$)
$Enc_{SK_{ccert}}(Cert_{CC})$
MakeCredential($SRK_{pub}, SK_{ccert}, CC_{name}$)
$Enc_{SK}(SK_{ccert}), DHK_{pub}$
$Sig_{CPS_{pr}}(CertInstallRes\{Enc_{SK_{ccert}}(Cert_{CC}),$
$Enc_{SK}(SK_{ccert}), DHK_{pub}\})$
TPM2_ActivateCredential(Load($CC$),
Load($SRK$), $Enc_{SK}(SK_{ccert}), DHK_{pub}$)
$SK_{ccert}$
Decrypt("AES", $SK_{ccert}, Enc_{SK_{ccert}}(Cert_{CC})$)
$Cert_{CC}$

Figure 5.13: Provisioning of Contract Credentials

key $SR_{ccert}$ is encrypted with a session key $SK$ derived using static-ephemeral ECDH (see [114], Section 24). For this, an ECDH keypair $\{DHK_{pub}, DHK_{pr}\}$ is generated and a seed $Z$ is calculated from $DHK_{pr}$ and $SRK_{pub}$. $Z$ is then used as input to KDFs to generate an encryption $SK$ and an HMAC key $HK$ (see Figure 5.12). Then, an HMAC is calculated over the encrypted data. Thereby, the value $CC_{name}$, i.e., a digest calculated over the objectAttributes of the public contract key (cf. Table 5.4), the defined policy of the OEM $Pol$, and $CC_{pub}$, is included in the entire process to guarantee the above-mentioned properties. Notably, the process is the same as used in TrustEV for encrypted the private contract key.

**Installation and Update of Contract Credentials.** The process of installing contract credentials uses the custom $CertInstallReq$ message and is detailed in Figure 5.13. The update process is executed in the same way, with the current contract credential of the EVCC used instead of the OEM provisioning credential in the request.

First, the EVCC generates a $CertInstallReq$ message containing its self-generated public contract key $CC_{pub}$ in the new field (cf. Section 5.5.1). It calls the TPM function `TPM2_Sign()` to load the OEM provisioning key and use $PC_{pr}$ for signing the message. The TPM first checks whether the defined policy $Pol$ is met (`PC_PolicyCheck()`) and then returns the signed $CertInstallReq$, which is forwarded by the SECC to the eMSP.

On receiving the *CertInstallReq* message the eMSP verifies the signature using $PC_{pub}$. If the eMSP does not support HIP, it ignores all corresponding changes and proceeds as specified by ISO 15118 (cf. Section 5.1.1). Otherwise, the

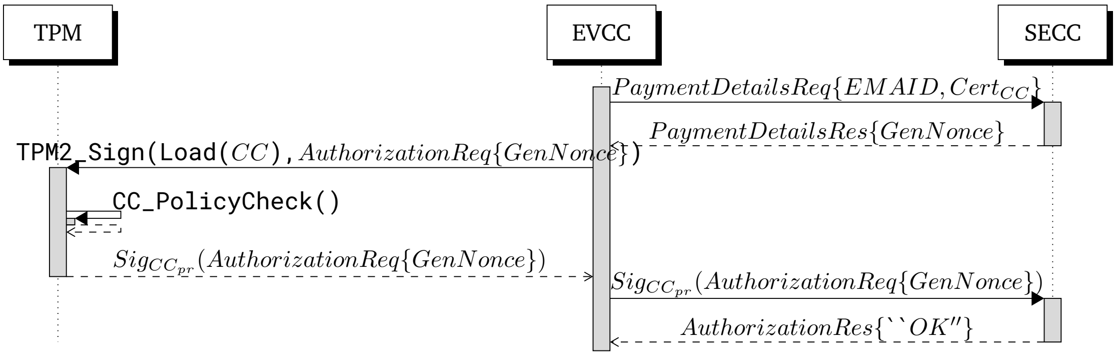

TPM
EVCC
SECC
$PaymentDetailsReq\{EMAID, Cert_{CC}\}$
$PaymentDetailsRes\{GenNonce\}$
TPM2_Sign(Load($CC$),$AuthorizationReq\{GenNonce\}$)
CC_PolicyCheck()
$Sig_{CC_{pr}}(AuthorizationReq\{GenNonce\})$
$Sig_{CC_{pr}}(AuthorizationReq\{GenNonce\})$
$AuthorizationRes\{``OK''\}$

Figure 5.14: PnC Authorization with Contract Credentials

eMSP generates the requested contract certificate and executes the credential protection protocol as described above in Section 5.5.2. The *CertInstallRes* message containing the encrypted $Cert_{CC}$, the encrypted $SK_{ccert}$, and the public $DHK_{pub}$ is signed with the private key $CPS_{pr}$ of the CPS and returned to the EVCC.

After validating the authenticity of the response, the EVCC can install the received contract credential by calling the function TPM2_ActivateCredential, which returns $SK_{ccert}$ necessary for decrypting $Cert_{CC}$ at last.

**Contract Credential Usage.** After activating its new contract credential, the EVCC can use the private contract key $CC_{pr}$ stored in the TPM for various V2G services. The general process for PnC authorization is shown in Figure 5.14, where $GenNonce$ is a nonce that needs to be signed with the private contract key $CC_{pr}$. The only difference to the ISO 15118 specification is that all PnC keys are stored in the TPM and sealed to a policy, i.e., additional operations are required before these keys can be used (in Figure 5.14 shown as generic CC_PolicyCheck()). Which conditions need to be satisfied in order to be able to use the contract key depends on the particular OEM policy substituting the generic *TPM2_PolicyAuthorize*. We discuss it in more details during the evaluation of HIP.

### 5.5.3. Security Discussion

Since HIP is based on the previously introduced TrustEV architecture, it inherits its security properties. Similarly, the security of the enhanced HIP solution strongly depends on the properties of the HSM used as a trust anchor and the required security goals can be achieved only when this HSM is compliant to the TCG specifications for TPM 2.0 and uses CC EAL4+ certified hardware qualified for automotive applications (cf. Section 5.4.3).

In contrast to TrustEV, our new ISO 15118 extension addresses a different provisioning strategy, where EVs are allowed to self-generate keypairs for their contract credentials. For this reason, during the following discussion of the security requirements defined in Section 5.3.1, we focus in particular on the requirements related to the secure generation of contract keys in an EV and secure enrollment of the vehicle using these keys at an eMSP. Notably, the analysis performed in Section 5.4.3 regarding security requirements from $RS_1$ to $RS_5$ also applies here.

$RS_1$ *Secure key storage:* With HIP, all keys are stored under protection of the certified TPM 2.0. As described in Section 5.5.1, all PnC keys carry the attributes *fixedTPM* and *fixedParent* preventing the export of the private keys. Thus, neither a backend hacker nor a car hacker can read out private keys, and HIP fulfills the security requirement $RS_1$.

$RS_2$ *Secure cryptographic operations:* Security requirement $RS_2$ is also fulfilled, since all cryptographic operations are performed by the TPM, with the private keys are only used in the shielded area of the TPM.

$RS_3$ *Key usage authorization:* HIP meets security requirement $RS_3$, by embedding an access policy *PolicyAuthorize* during generation of OEM provisioning and contract keys (cf. Section 5.5.1). Before any of these keys can be used, the TPM checks if the security policy is satisfied. As *PolicyAuthorize* is a “placeholder” policy, by setting it, for example, to a policy *TPM2_PolicyPCR*, access to these keys is sealed to predefined PCR values characterizing the firmware state of the EVCC and only possible in this state. However, runtime attacks still pose a security risk, because they target already loaded keys, i.e., after the successful policy assertion.

$RS_4$ *Secure key provisioning:* With HIP, all keys are generated directly in the TPM, thus, they are protected against relatively powerful adversaries (except for side-channel and runtime attacks). The symmetric key for contract certificate decryption is transferred encrypted for the EVCC's TPM. Thus, we consider this requirement fulfilled.

$RS_5$ *Cryptographic agility:* The TPM 2.0 is specified with a cryptographically agile interface allowing to change key length and cryptographic algorithms by simply providing a different input. However, this flexibility is limited in practice by the TCG's algorithm registry [116] and TPM chips available on the market. As discussed with regard to TrustEV, there is no hardware available supporting cryptographic needs of the new edition ISO 15118-20. The HIP solution does not introduce any cryptographic algorithms, because cryptographic services are provided by the TPM. Therefore, insecure cryptographic algorithms can be easily exchanged without affecting HIP itself. In deployed systems, this would either require updating the firmware of the TPM or, if this is not possible, replacing the TPM. Security requirement $RS_5$ is fulfilled.

$RS_6$ *Secure key generation:* OEM provisioning keypair $PC$ and contract credential keypair $CC$ are both generated by the TPM (using the TPM's random number generator) and stored within the protected key hierarchy (cf. Sections 5.5.2 and 5.5.2). Assuming the key generation of the used TPM is implemented securely, the security requirement $RS_6$ is fulfilled.

$RS_7$ *Trustworthy credential enrollment:* The EVCC can only decrypt and use its new contract certificate created by the eMSP for the requested contract key, only if the corresponding contract key has been generated in the same TPM as the SRK and OEM provisioning key, sealed to the defined policy (cf. certificate extension in Figure 5.4) and has the expected attributes (cf. Table 5.4). If any of the conditions fail, the EVCC cannot use the contract key in V2G services. Security requirement $RS_7$ is considered fulfilled.

### 5.5.4. Revocation of PnC Credentials

The proposed security architecture HIP addresses several aspects of secure key management [188], in the following we discuss the corresponding revocation aspects. In the context of ISO 15118, there are various reasons to revoke a PnC credential, such as sale of the vehicle, scrapping of the vehicle or its EVCC, EVCC's upgrade or repair, changes in one of the CAs or certificate policies, end of the service contract etc. The most critical reason to revoke a PnC credential is the compromise of its private key. The proposed solution is designed to protect all PnC credentials against this threat throughout their entire lifecycle. However, the security properties of HIP strongly depend on the security of the chosen HSM as we discussed above. Unfortunately, even certified security modules can suffer from implementation errors such as TPM-Fail [190] allowing car hackers to recover private ECDSA keys from some TPMs. Since TPM-Fail only affects ECDSA signature schemes, the root of the HIP key hierarchy SRK is not affected, because it is a decryption key. But as PnC credentials are used to perform ECDSA signatures in accordance to ISO 15118, it is important to consider available options to revoke these private keys with HIP.

ISO 15118 supports Online Certificate Status Protocol (OCSP) and Certificate Revocation Lists (CRLs) provided by the responsible SAs to announce the revocation of OEM provisioning and contract certificates. The standard does not address in detail how the SAs should process this information, only respective validations by the SECC during PnC session negotiation are required. As the EVCC uses its OEM provisioning credential to request new contract credentials at eMSPs, i.e., to prove that this request comes from a valid registered vehicle of the eMSP's customer, this creates an implicit dependency between these two credentials. After revoking an OEM provisioning credential, contract certificates requested on its basis may still remain valid, unless the eMSP actively collects and checks CRLs from different OEMs using registered PCIDs of its customers' vehicles. While the eMSP can directly revoke contract certificates issued by its CA, when an OEM provisioning certificate is revoked extra communication between the OEM and the eMSP is required.

With HIP, it is possible for the OEM to revoke both credentials at the same time, because their keys reside in the same TPM. In this case, the EVCC can simply generate a new contract key and request a new contract certificate as described in Section 5.5.2. ISO 15118 does not include a concept for installing new OEM provisioning credentials, because this credential serves as a long-term vehicle identity and needs to be installed in a trusted environment and also because in the past the OEM was not involved into V2G services. With the introduction of the SRK as root of the key hierarchy in HIP, however, the same installation method as used for contract certificates can be easily applied to receive new OEM provisioning certificates as long as the SRK can be trusted. Only if the root of the TPM hierarchy is compromised, the EVCC needs to be re-initialized as described in Section 5.5.2. In this case, the certificates that require revocation can be identified by their HIP extension, which includes the $SRK_{pub}$.

### 5.5.5. Evaluation

In this section, we describe the PoC implementation of our new security architecture HIP and evaluate the prototype against the functional requirements defined in Section 5.3.2.

**Proof-of-Concept Implementation.** The HIP prototype is realized on the test-bed setup first developed for TrustEV PoC and described in Section 5.4.4. Using this setup, the HIP protocol extensions are implemented on the development boards representing EVCC and SECC as described in Section 5.5.1 and Section 5.5.2. The backend functionality is also implemented on the SECC hardware. We employ the TPM 2.0 chip Infineon OPTIGA TPM SLI 9670 [191] possessing all necessary certifications in our tests to provide reliable results with regard to the solution's feasibility. The private keys $PC_{pr}$ and $CC_{pr}$ are sealed with the TPM policy *PolicyAuthorize*, the OEM provides a signed policy with the `TPM2_PolicyPCR` command, and the signature validation ticket is pre-calculated as explained further. The EVCC's firmware state is recorded during a measured boot and stored in one of the TPM's PCRs. For simplicity, we omit the measured boot in our implementation and instead hash the contents of a file, representing the firmware, into the selected PCR. The EVCC loads the signed $Pol_{PCR}$ digest provided by the OEM together with the current firmware and verifies the signature using the OEM's public key. The result is a *ticket*, proving the successful signature verification by the TPM. For better performance, the EVCC may acquire and store the ticket from its TPM in advance, thus, omitting the costly asymmetric operation in the credential usage process. The EVCC starts a policy session and executes the `TPM2_PolicyPCR` command. The intermediate policy digest should be equal to $Pol_{PCR}$ signed by the OEM. Afterwards, the EVCC executes the `TPM2_PolicyAuthorize` command, providing the verified $Pol_{PCR}$ and the ticket. Only if the intermediate policy is equal to $Pol_{PCR}$, i.e., the firmware state is as expected, is the assertion successful and the EVCC's contract key can be used. Generating and encrypting $Cert_{CC}$ by the eMSP is done in Java using Bouncy Castle[14].

**Analysis of Functional Requirements.** In order to verify if the HIP prototype fulfills its design goals, we perform measurements using our test-bed setup and compare the results with the default implementation RISE V2G[15], whereby all measurements and baselines refer to the version ISO 15118-2 [38]. We analyze our findings with respect to the functional requirements below.

$RF_1$ *Minimal overhead:* To show $RF_1$ is fulfilled, we analyzed the communication and computational overhead. The goal is to meet the constraints defined by ISO 15118. With respect to the communication overhead, the contract certificate data field in the response is limited to 800 bytes and the encrypted private contract key field is limited to 48 bytes. In the HIP prototype, the encrypted certificate including the IV is transmitted in the certificate field and requires only 608 bytes. Of course, this value depends on the original size of the contract certificate and may vary if the eMSP, e.g., fills the complete allowed size with internal information, but eMSPs supporting HIP can adjust accordingly. The encrypted symmetric AES key $SK_{ccert}$ is transmitted in the encrypted contract key field, and requires 40 bytes or 52 bytes depending on the hash function[16]. Thus, this constraint is only met if SHA-1 is used during the symmetric key encryption.

The computational overhead shall not result in exceeding timeouts defined by ISO 15118 (cf. [38], Table 109). The maximum time between an EVCC's $CertInstallReq$ message generation and the corresponding response is 4500 ms. Thus, the additional operations required by our protocol extensions at the eMSP should not be too time-consuming. We repeated measurements 100 times using Java's System.nanoTime(). On average, it took 452.5 ms between $CertInstallReq$ and eMSP's response, with the maximum value never being exceeded. Though our test-bed does not simulate round trip times for backend communications, and the above overhead incurred solely by HIP, the eMSP saves a key generation operation on its side and compared to ISO 15118 only needs to perform an additional symmetric encryption operation. Thus, with HIP eMSPs generating contract credentials "on the fly" might actually win some time. The eMSPs preferring to pre-generate the credentials as described in VDE-AR 2802 might have problems with this approach. Our protocol extension also introduces additional overhead by using the TPM from the EVCC, which falls within the EVCC sequence performance time of 40000 ms. We measured the time for the entire certificate provisioning process, between the start of signing the request and the end of decrypting the received $Cert_{CC}$ (cf. Fig. 5.13) 100 times. The maximum value was never reached and the

[14] `https://www.bouncycastle.org/java.html`
[15] `https://github.com/V2GClarity/RISE-V2G`
[16] 20 bytes HMAC with SHA-1 or 32 bytes HMAC with SHA256 and 18 bytes encrypted ($size\{SK_{ccert}\}||SK_{ccert}$) (cf. Figure 5.12).

average value was with 5385.1 ms far below the maximum value. In addition, we have measured the time for signing a PnC authorization request with the TPM using $CC_{pr}$ (including the policy assertion) individually, which was 753.3 ms.

$RF_2$ *ISO 15118 conformance:* Our protocol extension alters the ISO 15118 protocol to increase the credential provisioning security. The certificate request message is extended with a public contract key (cf. Figure 5.11) and a minor protocol version number is suggested. We also redefined the fields in the certificate response message in our PoC to transfer alternative parameters. This may lead to incompatibilities if any of the intermediate actors verify the format of these messages. Moreover, it also contradicts the strongly typed XML structure of ISO 15118 protocol messages. In order to integrate HIP into ISO 15118 the respective types should be added to the specification under the new minor protocol version. The protocol message flow remained unchanged, though we alter the logic: the eMSP now has to wait for the request from the EVCC with a new contract key before it can decide to issue a new certificate, which can become a disadvantage, e.g., for revocations. Getting a bit ahead, we can report that these reasons eventually lead to rejection of our proposal by the standardization committee. Since these changes are required to substantially increase the security, we nevertheless consider the functional requirement $RF_2$ fulfilled.

$RF_3$ *Backwards compatibility:* Since the introduced certificate extension is non-critical and eMSPs can ignore any additional information provided by the EVCC and proceed as usual, this functional requirement is fulfilled.

**Backend Communication Aspects for ISO 15118 Protocol Extensions.** Since the security architecture HIP substantially alters message formats, it is important to consider how these changes influence the communication path between the SECC of the Charge Point and the eMSP, because even if we succeed to officially integrate our proposals into the new edition of ISO 15118, the rest of the infrastructure needs to remain compatible. In our system model in Section 5.1, we omitted the aspects that are out of scope for ISO 15118. Therefore, we shortly analyze them below. The de facto communication standard between a Charge Point and backend systems (usually via its CPO) is the OCPP [40]. In the newer editions starting from OCPP 2.0 [130], the protocol integrates ISO 15118-2 messages and forwards certificate installation and update requests as Base64 encoded strings using the *Get-/Updat15118EVCertificate* messages. As long as no intermediate parsing is performed, the OCPP is not affected by HIP except for the performance overheads. The older but more prevalent version OCPP 1.6 [80] does not natively support ISO 15118, but it offers a flexible format with *DataTransfer* messages that can contain arbitrary data. Therefore, the above conclusion also applies here.

If the CPO does not assume the role of the eMSP for the EV's user, the CPO has to forward the EVCC's request to a clearing house or directly to the respective eMSP. None of the protocols specified for this purpose, e.g., Open Clearing House Protocol (OCHP) 1.4 [109], Open Intercharge Protocol (OICP) 2.2 [192], OCHPdirect 0.2 [110] or Open Charge Point Interface (OCPI) 2.2 [111], currently provides a concept for the transmission of ISO 15118 certificate installation and update messages. Therefore, either these messages are forwarded as a whole similar to OCPP or the respective actor only extracts the data necessary to perform its role. For the latter, any data that the actor does not recognize should be ignored in the same way as it is specified for ISO 15118. If it is the case, the HIP functionality cannot be implemented anymore and the due to backward compatibility the provisioning proceeds according to the original specification.

### 5.5.6. HIP Summary

In this section, we presented the concept, prototypical implementation and evaluation of HIP, an HSM-based security architecture for secure generation of PnC credentials in the EV and trustworthy enrollment at an eMSP with self-generated contract keys. With HIP, we address all shortcomings of ISO 15118 regarding credential management, by considering secure generation, storage, provisioning/enrollment, usage and revocation of OEM provisioning and contract credentials. In contrast to TrustEV, the HIP solution alters the standard provisioning approach and allows eMSPs to reliably outsource critical cryptographic key generation to the vehicles of their customers if these vehicles are equipped with an HSM offering the required security guarantees. The V2G infrastructure is only used to request a new contract certificate for public contract keys, with their private keys never leaving the protected environment of the chosen HSM. We defined the HIP components for an HSM-equipped EVCC and the backend systems of its OEM and eMSP as well as the necessary extensions to the ISO 15118 protocol flow. In order to enable the binding between the vehicle's HSM and the generated keys, OEM provisioning and contract certificates use the optional TrustEV extension. We use the openly specified TPM 2.0 standard for implementing the HSM interface for secure key generation and storage, credential protection, enhanced authorization and cryptographic operations, because the corresponding open-source software and certified hardware are readily available for reproducible benchmarks.

HIP protects private keys of all PnC credentials against car hackers and backend hackers, except for runtime and side channel attacks, which are considered too expensive compared to the potentially gained profit. Moreover, HIP provides additional security guarantees to the eMSP, provably deploying the contract credential on the EVCC it was intended for, which is not reachable with the standard ISO 15118 approach. It ensures backward compatibility, i.e., V2G actors, which do not support the extension, can still use the regular ISO 15118 mode. The respective protocol changes are not critical and the introduced overhead is still within the constraints defined by ISO 15118 as shown by HIP's PoC.

There are two main disadvantages with regard to the ISO 15118 ecosystem: (i) HIP requires additional parameters to be exchanged between the EVCC and the eMSP, which cannot be incorporated into existing message format, like it is the case with TrustEV; (ii) by requiring the public key being provided by the EVCC, HIP hinders pre-generation of contract credentials by eMSPs in accordance to VDE-AR 2802. For these reasons, the proposal to integrate the HIP solution in the upcoming edition of ISO/FDIS 15118-20 [39] was not accepted for the current revision round by the standardization committee ISO/TC 22/SC 31/JWG 1 responsible for the development of the ISO 15118 standard series. In the next section, we address the mentioned disadvantages and investigate an approach, where the EVCC's public contract keys can be securely distributed among interested eMSPs using the Global Certificate Store (GCS) specified in VDE-AR 2802 [78].

## 5.6. HIP 2.0: Integration of HSM-based Identities into the Plug-and-Charge Ecosystem

In this section, we introduce our new security extension HIP 2.0 that enables secure provisioning of PnC credentials, where contract certificates for EV self-generated contract keys are created in a distributed and asynchronous manner according to the guideline VDE-AR 2802 [78]. This extension is the further development of the previously presented security architecture HIP (cf. Section 5.5). While HIP focuses on the secure and verifiable generation of contract credential keys in the vehicle's HSM and the trustworthy enrollment of the vehicle with these keys at an eMSP, HIP 2.0 improves this solution by supporting established CA processes such as using CSRs [184], out-of-band credential provisioning via channels outside of the ISO 15118 communication, and the revised certificate update process of the upcoming edition ISO 15118-20. As its successor, HIP 2.0 aims to protect PnC credentials from car hackers and backend hackers (cf. Section 5.2), whereby also considering backend systems participating in the out-of-band provisioning beyond the eMSP (e.g., CPS and GCS, cf. Section 5.1).

HIP 2.0 similarly relies on a certified HSM for secure cryptography, generation and storage of cryptographic keys on the vehicle's EVCC to meet the security requirements from Section 5.3.1 and achieve the required Level of Assurance (LoA) (cf. [187]). In order to provide an assurance to the eMSP that the keys belonging to a new contract credential are generated and stored in the vehicle's HSM and are adequately secured, HIP 2.0 adopts the credential protection feature of HIP, with several adaptations for the distributed and asynchronous credential provisioning. These include changes to the encryption and credential activation steps (see Section 5.6.3). The API for the HSM is defined in compliance with the open TCG standard for TPM 2.0 [114].

While the ISO 15118 standard series focuses on the communication between EV and Charge Point, the application guideline VDE-AR 2802 [78] addresses the backend aspects regarding asynchronous generation and delivery of credentials. In order to enable the proposed security features in ISO/DIS 15118-20, changes are required in the initial preparation of the EV performed by the OEM, in the generation of the contract keypair, during creation of contract certificates for EV enrolling with own contract keys, OEM provisioning and contract certificate profiles, and the charging process to make use of the TPM. Since the credentials are provisioned using VDE-AR 2802, changes to the data stored in the pools of GCS and some auxiliary processes are also required. ISO/DIS 15118-20 does not allow for the transfer of EVCC's public keys or CSRs, thus, HIP 2.0 alters the message format to exchange the security information, but the message sequencing remains unchanged. In order to verify that this new extension does not affect the core PnC functionality, we evaluate its security and performance by integrating HIP 2.0 into our TrustEV/HIP proof-of-concept implementation and test-bed setup.

In summary, we make the following main contributions:

- We present HIP 2.0, a new security architecture design extending HIP [7] to provide beside secure and verifiable generation, storage and usage of PnC credentials in vehicle's HSM and trustworthy credential enrollment at an eMSP, the support of standard certificate issuance procedures using CSRs and out-of-band provisioning via the application guideline VDE-AR 2802.
- We provide a concept for integrating the TPM 2.0 credential protection feature into the processes of VDE-AR 2802 and ISO/DIS 15118-20, in a nearly compliant and backward compatible manner.
- We analyze and discuss the security properties of HIP 2.0 considering strong adversaries with control over the EVCC, communication channels, and backend systems.
- We evaluate critical performance aspects of HIP 2.0 and discuss the feasibility of the solution under ISO/DIS 15118-20 constraints.

This section is organized as follows: In Section 5.6.1, we describe the components of the new security extension HIP 2.0. In Section 5.6.2 and in Section  5.6.3, we define the changes to the provisioning steps in ISO/DIS 15118-20 and in VDE-AR 2802 respectively related to the proposed extension. Section 5.6.4 provides the discussion on achieving the design goals with HIP 2.0 regarding security, performance and compliance to the standards. Finally, Section 5.5.6 provides a summary of the obtained results.

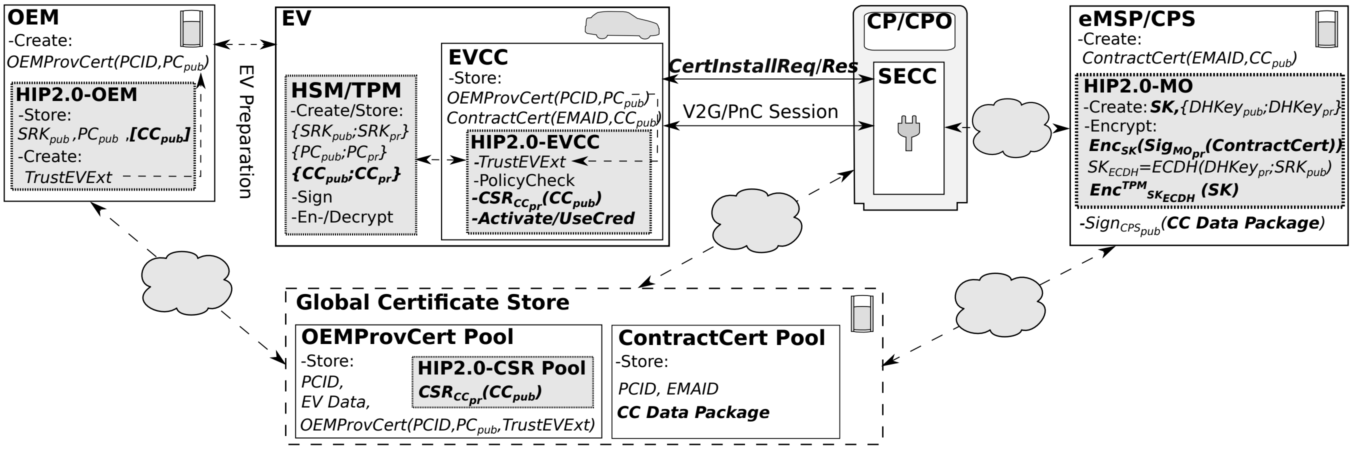


Figure 5.15: HIP 2.0 Security Architecture

### 5.6.1. HIP 2.0 Components

The HIP 2.0 security architecture consists of a new backend component `HIP2.0-CSR Pool` in the GCS and the existing HIP components for the OEM, EVCC, and eMSP, to which new functions are added to support out-of-band credential provisioning as shown in Figure 5.15. The `HIP2.0-OEM` component, beside initiating the TPM key hierarchy with keypairs $\{SRK_{pub}, SRK_{pr}\}$ and $\{PC_{pub}, PC_{pr}\}$, setting key access policy $Pol$, and issuing the OEM provisioning certificate $Cert_{PC}$ with the custom security extension makes these data available for upload into the provisioning certificate pool of the GCS. As we show later, the *EV Preparation* step is performed periodically via the proprietary telematics link between the OEM and the EVCC, in order to keep the EV credential information in the GCS up-to-date.

The `HIP2.0-EVCC` component uses the vehicle's TPM to generate keypairs for contract credentials $\{CC_{pub}, CC_{pr}\}^1$, ..., $\{CC_{pub}, CC_{pr}\}^n$ using the profile from Figure 5.16 and the OEM policy and stores them underneath the SRK in the TPM key hierarchy. After a keypair is available, this component creates a CSR for the respective public key and calls the TPM to sign this CSR. This requires a correct assertion of the private key's policy. These CSRs can be provided to `HIP2.0-OEM` to be delivered to the GCS or be used to directly request a new contract certificate $Cert_{CC}$ from an eMSP. Thereby, a PKCS #10 CSR [193] format is used according to the common CAs practice[17]. In order to include the CSR into $CertInstallReq$, `HIP2.0-EVCC` creates a modified message with an extra field. The response is also modified, and, therefore, also processed by this component. Other functions are inherited from HIP without change (cf. Section 5.5.1).

The `HIP2.0-MO` component in the eMSP/CPS backend (can be one or different actors) recognizes based on the custom extension of the OEM provisioning certificate and the presence of a CSR (i.e., field size) in the installation request that the EVCC supports HIP 2.0. Now, two options are possible: (i) `HIP2.0-MO` follows VDE-AR 2802 and queries the GCS to check whether there are pre-generated contract certificate data packages available and proceeds as described in Section 5.6.3; (ii) `HIP2.0-MO` similar to HIP generates a new contract certificate data package "on-the-fly". While issuing a contract certificate, whether for the direct or out-of-band delivery, `HIP2.0-MO` uses the HIP function for credential protection but applies it only to the signature of this certificate. This adjustment is necessary to keep the certificate itself readable so that it can be mapped to a respective TPM keypair by the `HIP2.0-EVCC` to decrypt the signature. The credential protection ensures that the complete certificate can be used only by the EVCC with the access to the HSM storing the SRK, the OEM provisioning key and the contract key with the matching attributes. Therefore, compared to ISO/DIS 15118-20, the eMSP needs to implement an additional encryption method and perform two cryptographic operations. The contract certificate data package is then loaded into the GCS according to VDE-AR 2802.

HIP 2.0 requires changes in the ISO/DIS 15118-20 certificate installation request message, a custom X.509v3 certificate extension, and some adaptions of the ISO/DIS 15118-20 and VDE-AR 2802 protocol flow for provisioning contract credentials. We discuss these changes below.

**Cryptographic Key Profiles.** With HIP 2.0, all keys used for PnC are generated inside the EVCC's TPM. Figure 5.16 shows the TPM keys used by HIP 2.0: the OEM provisioning credential keypair $\{PC_{pub}, PC_{pr}\}$, the contract credential

[17] The certificate policy for the V2G PKI of Hubject (the most established CA for EV charging [194]) supports only two methods for credential provisioning [195]: (i) with key generation by the CA and (ii) via a PKCS #10 CSR.

| Attribute | $SRK$ | $PC$ | $CC$ |
|---|---|---|---|
| type: | TPM2_ALG_ECC | | |
| nameAlg: | TPM2_ALG_SHA256 | | |
| object-Attributes: | fixedTPM, fixedParent, restricted, decrypt, sensitiveDataOrigin, userWithAuth, noDA | fixedTPM, fixedParent, sign, decrypt, sensitiveDataOrigin | fixedTPM, fixedParent, sign, sensitiveDataOrigin |
| authPolicy: | n/a | TPM2_PolicyAuthorize | |
| curveID: | TPM2_ECC_NIST_P256 | | |

Figure 5.16: Public Area Attributes of the TPM 2.0 Keys for ISO/DIS 15118-20

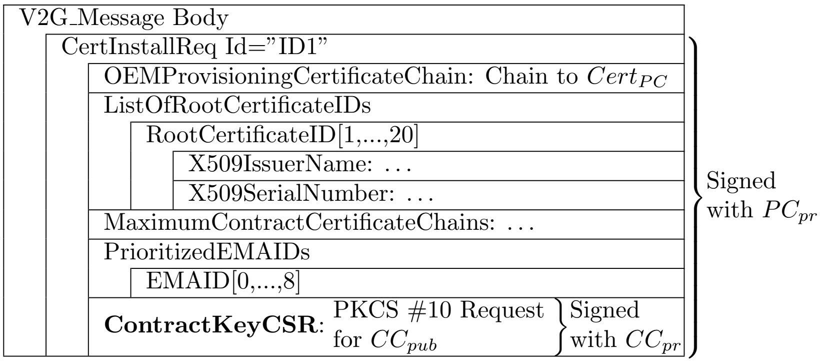

V2G_Message Body
CertInstallReq Id="ID1"
OEMProvisioningCertificateChain: Chain to $Cert_{PC}$
ListOfRootCertificateIDs
RootCertificateID[1,...,20]
X509IssuerName: ...
X509SerialNumber: ...
MaximumContractCertificateChains: ...
PrioritizedEMAIDs
EMAID[0,...,8]
**ContractKeyCSR**: PKCS #10 Request for $CC_{pub}$
Signed with $CC_{pr}$
Signed with $PC_{pr}$

Figure 5.17: Extension in ISO/DIS 15118-20 $CertInstallReq$ Message

keypair $\{CC_{pub}, CC_{pr}\}$, and a TPM storage root key $\{SRK_{pub}, SRK_{pr}\}$. The only change compared to HIP is that contract keys do not carry *decrypt* attribute anymore, because starting with ISO/DIS 15118-20 [183] certificate updates are deprecated, and all keys use SHA256 as *nameAlg* hash function.

**Contract Certificate Request.** In order to provide PKCS #10 CSRs with the newly generated public contract keys to an eMSP, we extend the definition of the $CertInstallReq$ message with the optional field *ContractKeyCSR* encoded as a Base64 string as shown in Figure 5.17. The CSR is signed with $CC_{pr}$ according to RFC2986 [193] providing a proof-of-possession of this private contract key. Since the EVCC does not know the correct EMAID at the moment of the CSR generation, the subject name field is omitted and added in the contract certificate by the eMSP's CA.

**Custom Certificate Extension.** HIP 2.0 uses the custom TrustEV certificate extension in the OEM provisioning certificate (see Figure 5.4) to provide the necessary security information about the EVCC's TPM to an eMSP. This information includes the SRK of the TPM key hierarchy and the digest of the OEM key access policy necessary for the credential protection protocol [114].

### 5.6.2. HIP 2.0 Protocol Extensions

In the following, we describe how new security functionality provided by HIP 2.0 can be integrated into PnC credential provisioning (cf. Section 5.1.1). Since HIP 2.0 extends the previously introduced HIP architecture, some steps remain mostly unchanged compared to the description in Section 5.5.2. Therefore, we focus only on the additional modifications. Notably, the only HIP 2.0 change related to ISO/DIS 15118-20 is the simplification of certificate updates, which in this edition also performed using certificate installation requests and OEM provisioning certificate[18].

**EV Preparation.** This step is performed exactly like it is defined for the HIP extension (cf. Section 5.5.2). Before delivery of the vehicle to its user, the OEM creates the protected key hierarchy in the EVCC's TPM according to the templates

[18] In ISO/FDIS 15118-20, processes regarding PnC authentication mostly remain unchanged, only message formats are slightly reworked and TrustEV [1] is officially integrated into the standard.

from Section 5.6.1, with the SRK as the root and the asymmetric OEM provisioning key $PC$ sealed to an extendable access policy *PolicyAuthorize* as its child. Optionally, contract keys can also be created at this step. The manufacturer then generates the OEM provisioning certificate $Cert_{PC}$ including the custom extension containing $SRK_{pub}$ and policy digest $Pol$ (cf. Section 5.6.1). The eMSP extracts the information from this certificate extension to encrypt the signature of a new contract certificate as described below.

**Generation of Contract Credentials.** In contrast to HIP, after generating a new contract keypair $\{CC_{pub}, CC_{pr}\}$ with the help of its TPM, the EVCC additionally creates a CSR for the newly generated public contract key and calls its TPM to sign it with the respective private key in accordance with RFC 2986 [193]. Before using the contract key, the TPM validates the access policy defined by the OEM for this key.

The second part of the contract credential generation, i.e., issuing a contract certificate $Cert_{CC}$, is performed by the eMSP on receiving the custom certificate installation request $CertInstallReq$ containing the CSR from the EVCC. The process consists mostly of the same steps as for HIP (cf. Section 5.5.2), with the following two differences. First, the newly generated contract certificate does not receive the custom extension, because certificate updates are deprecated starting with ISO/DIS 15118-20. Second, the credential protection protocol (cf. [114], Section 24) is executed only with the signature of the newly generated certificate, not the complete certificate. The sequence diagram of this protocol is shown in Figure 5.18. The certificate's signature is encrypted using AES and a newly generated symmetric key $SR$ (cf. $Enc_{SK}(Cert_{CC})$ in Figure 5.18), with the initialization vector IV randomly generated before encryption. Afterwards, the encryption key $SR$ is wrapped as follows (cf. $Enc_{SRK_{pub}}(SK)$ in Figure 5.18): session key $SK_{ECDH}$ for the encryption is derived using static-ephemeral ECDH (see [114], Section 24). For this, an ECDH keypair $\{DHKey_{pub}, DHKey_{pr}\}$ is generated and a seed $Z$ is calculated from $DHKey_{pr}$ and $SRK_{pub}$. $Z$ is then used as input to KDFs to generate the session key $SK_{ECDH}$ and an HMAC key (see Figure 5.12). Then, an HMAC is calculated over the encrypted data. In order to bind this encrypted data to the contract key in the EVCC's TPM, a digest calculated over the attributes of the public contract key, the OEM policy, and public key itself, is included during the KDF and HMAC calculation.

**Installation of Contract Credentials.** The process of installing contract credentials with HIP 2.0 is detailed in Figure 5.18. In the first step, the EVCC generates a modified $CertInstallReq$ message containing $Cert_{PC}$ with the custom X.509v3 extension and the signed CSR for its (new) public contract key in the additional field (cf. Section 5.6.1). It calls the TPM with the `TPM2_Sign()` function to load $PC$ and use $PC_{pr}$ for signing the request message. The TPM checks whether the defined policy is met (`PC_PolicyCheck()`) and then returns the signed request. The EVCC sends the signed request to the SECC within an active PnC session, which forwards it to the eMSP.

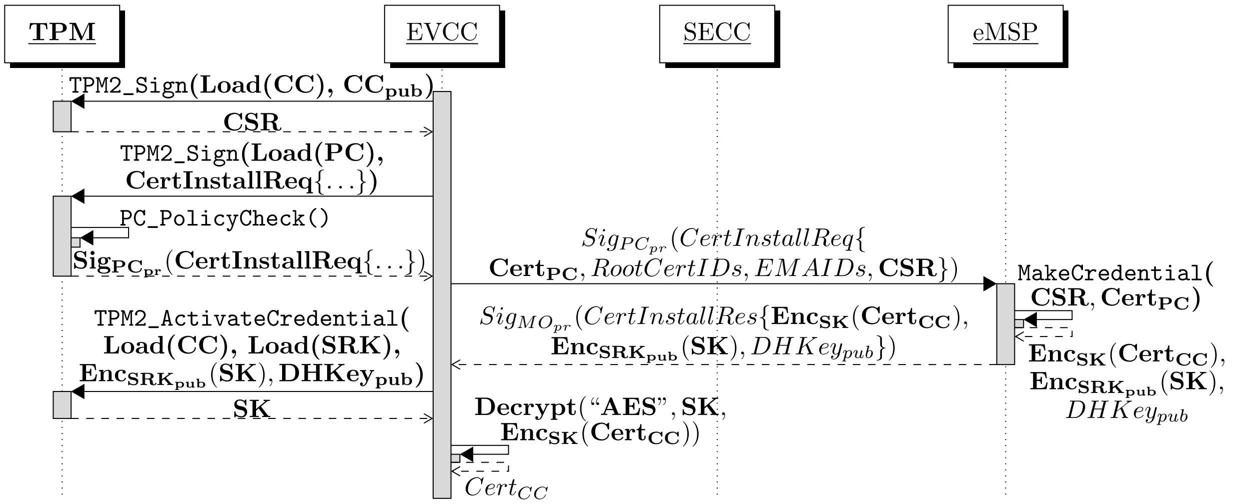

TPM
EVCC
SECC
eMSP
TPM2_Sign(Load(CC), CC$_{pub}$)
CSR
TPM2_Sign(Load(PC), CertInstallReq{...})
PC_PolicyCheck()
Sig$_{PC_{pr}}$(CertInstallReq{...})
$Sig_{PC_{pr}}(CertInstallReq\{$ $Cert_{PC}, RootCertIDs, EMAIDs, CSR\})$
MakeCredential( CSR, Cert$_{PC}$)
Enc$_{SK}$(Cert$_{CC}$), Enc$_{SRK_{pub}}$(SK), $DHKey_{pub}$
$Sig_{MO_{pr}}(CertInstallRes\{Enc_{SK}(Cert_{CC}),$ $Enc_{SRK_{pub}}(SK), DHKey_{pub}\})$
TPM2_ActivateCredential( Load(CC), Load(SRK), Enc$_{SRK_{pub}}$(SK), DHKey$_{pub}$)
SK
Decrypt("AES", SK, Enc$_{SK}$(Cert$_{CC}$))
$Cert_{CC}$

Figure 5.18: Credential Provisioning over ISO/DIS 15118-20

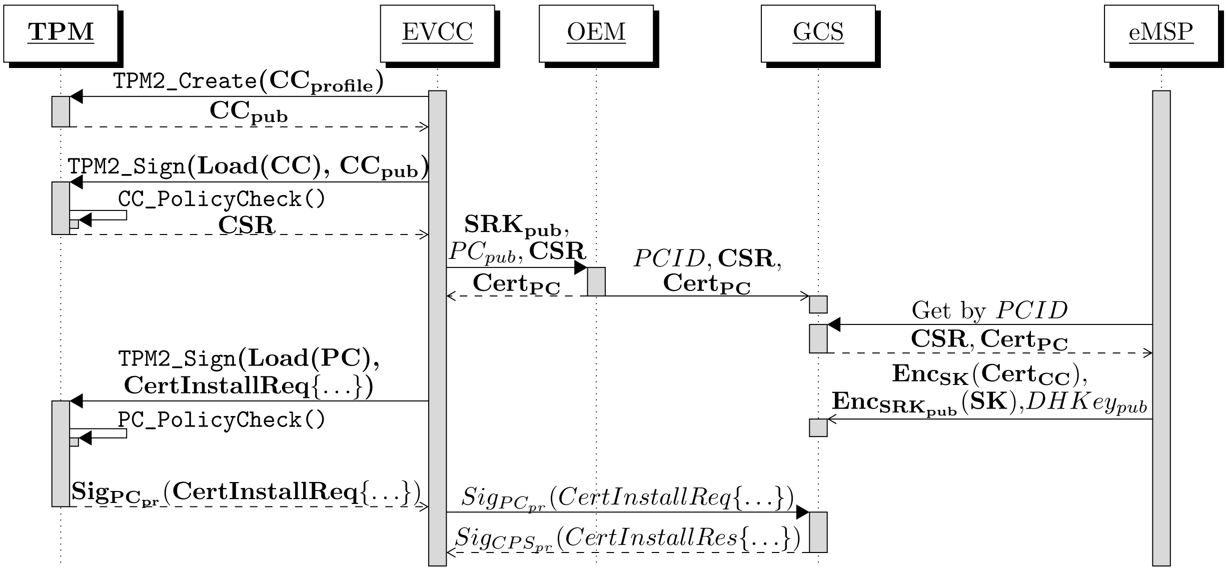
On receiving the $CertInstallReq$ message, the eMSP verifies the signature using $PC_{pub}$ provided in this request to ensure that the request comes from a registered vehicle of its customer. If the eMSP does not support any security extensions, it ignores all modifications and follows the usual approach of generating a new contract keypair with a corresponding contract certificate and sending it to the EVCC. If the eMSP supports the HIP 2.0 extension, it extracts the CSR from the additional field of the certificate request and verifies the signature of the CSR using the included contract key $CC_{pub}$ (aka proof-of-possession). If this verification step succeeds, the eMSP issues a new certificate $Cert_{CC}$ as described above. The modified $CertInstallRes$ message containing the certificate with the encrypted certificate $Enc_{SK}(Cert_{CC})$, the encrypted $SK$, and the public $DHKey_{pub}$ is signed with the private key $MO_{pr}$ of the eMSP (if eMSP takes the role of CPS, otherwise, it is signed with $CPS_{pr}$ of the CPS) and returned to the EVCC via the SECC.

On receiving the response, the EVCC verifies the source and calls the TPM with the `TPM2_ActivateCredential` to load the keys $CC$ and $SRK$. The TPM uses $SRK_{pr}$ to derive the ECDH session key and decrypt $SK$, and then returns $SK$ to the EVCC that can in turn decrypt the signature of $Cert_{CC}$ using AES. The TPM can only perform the decryption operation, if both keys the $SRK_{pr}$ and the $CC_{pr}$ matching the public key parameters provided in the request (key attributes, public key value and access policy) are loaded at the same time, thus, enforcing the required binding between these entities and guaranteed security controls.

**Contract Credential Usage.** After decrypting the signature of $Cert_{CC}$, the EVCC can use the private contract key $CC_{pr}$ stored in the TPM for V2G services including PnC. For instance, for PnC authorization the EVCC needs to provide its valid contract certificate to the SECC to start challenge-response authentication protocol using its contract key. Only if the signature on the challenge sent by the SECC can be verified with $CC_{pub}$ from $Cert_{CC}$ and the $Cert_{CC}$ chain can be traced up to a trusted root CA, which requires the signature of $Cert_{CC}$ to be verifiable, the PnC can be activated. The only difference of HIP 2.0 in the PnC authorization process is that creating a signature with $CC_{pr}$ requires interaction with the TPM as well as the correct assertion of the key's policy. The same change is introduced in all other ISO/DIS 15118-20 processes that use the private contract key, e.g., signing of metering confirmations.

### 5.6.3. Integration of HIP 2.0 in VDE-AR 2802

In the following, we describe changes in the processes related to the application guideline VDE-AR 2802 [78], which are necessary to enable out-of-band provisioning of contract credentials with security properties of HIP 2.0. We go through the steps defined in Section 5.1.2 and discuss the proposed adjustments shown in Figure 5.15 and Figure 5.19.



TPM
EVCC
OEM
GCS
eMSP
TPM2_Create($\mathbf{CC_{profile}}$)
$\mathbf{CC_{pub}}$
TPM2_Sign($\mathbf{Load(CC), CC_{pub}}$)
CC_PolicyCheck()
CSR
$\mathbf{SRK_{pub}}$,
$PC_{pub}$, CSR
$\mathbf{Cert_{PC}}$
$PCID$, CSR,
$\mathbf{Cert_{PC}}$
Get by $PCID$
CSR, $\mathbf{Cert_{PC}}$
$\mathbf{Enc_{SK}(Cert_{CC})}$,
$\mathbf{Enc_{SRK_{pub}}(SK)}$,$DHKey_{pub}$
TPM2_Sign($\mathbf{Load(PC)}$,
CertInstallReq{...})
PC_PolicyCheck()
$\mathbf{Sig_{PC_{pr}}}$(CertInstallReq{...})
$Sig_{PC_{pr}}(CertInstallReq\{\ldots\})$
$Sig_{CPS_{pr}}(CertInstallRes\{\ldots\})$

Figure 5.19: Credential Provisioning over VDE-AR 2802

**EV Preparation.** In addition to uploading the vehicle data and the OEM provisioning certificate into the GCS queried by eMSPs to pre-generate contract credentials for its customers' vehicles, the OEM needs to ensure that CSRs created for the provisioning with HIP 2.0 are available to eMSPs before a certificate installation request is sent in a PnC session. Therefore, a new CSR pool is added to the GCS as part of the OEM provisioning certificate pool to store these data.

Instead of generating a contract keypair and a CSR directly before sending a certificate request to an eMSP, the EVCC's TPM has to pre-generate multiple keypairs $\{CC_{pub}, CC_{pr}\}$ and sign CSRs for the corresponding public keys. This step is necessary to ensure that there are always sufficient public keys available for a particular vehicle to support different use cases, e.g., revocation and contracts with different eMSPs. The OEM then uses the proprietary telematics link to the EVCC to periodically read out the CSRs and upload them to the GCS. The EVCC may also actively provide new CSRs to its OEM via this link. Since telematics systems are commonly available in modern vehicles, this requirement is not limiting. For example, Ford Motor Co. announced that they are planning to use a telematics-based method to provision ISO 15118 contract credentials [194].

**Issuance of Contract Credentials.** After the conclusion of a contract between an EV user and the eMSP, the eMSP uses the provided PCID to query GCS and extract the $Cert_{PC}$ and a CSR. The CSR is then deleted from the GCS so that each keypair is used only once to issue a certificate. The eMSP's CA validates the CSR and generates the corresponding contract certificate $Cert_{CC}$. Afterwards, the eMSP's ISO 15118 handler encrypts the signature of $Cert_{CC}$ using the TPM information $SRK_{pub}$ and $Pol$ from $Cert_{PC}$ as described in Section 5.6.2.

The eMSP builds a contract certificate data package as per [78], where a contract certificate with encrypted signature replaces a plaintext contract certificate and the credential protection information – encrypted private key. The eMSP sends this data package to the CPS for verification and signing. The CPS saves the contract certificate data packages provided by all eMSPs for a particular vehicle under the PCID of this vehicle in the contract certificate pool of the GCS.

**Installation of Contract Credential.** According to VDE-AR 2802, the distribution of contract certificates always happens via the CPS checking out the pre-generated contract certificate data packages from the certificate store. Therefore, when the SECC receives from the EVCC a custom $CertInstallReq$ including a CSR and an OEM provisioning certificate with TrustEV extension (cf. Section 5.6.1), it forwards it to the CPS. The CPS extracts the PCID from the request and downloads all contract certificate data packages available for this PCID from the respective pool of the GCS. If this operation is successful, the CPS ignores the CSR from the $CertInstallReq$ and uses the pool data instead. The signed data packages are written into $CertInstallRes$ messages (one per certificate) and returned to the EVCC via the SECC. Otherwise, the request may be forwarded to the eMSP and the installation proceeds according to HIP.

With out-of-band provisioning, the delivered certificate(s) with the encrypted signature (cf. $Enc_{SK}(Cert_{CC})$) does not necessarily match the CSR provided by the EVCC in the request. Therefore, in order to decrypt the signature and be able to use the contract certificate, the EVCC needs to determine to which of its contract keys this certificate belongs. For this purpose, the EVCC can use the Subject Key Identifier extension of the provided certificate for identifying the associated contract keypair. This extension is mandatory in ISO/DIS 15118-20 and is calculated in accordance with RFC 5280 (cf. [184], Section 4.2.1.2, Method (2)). After the EVCC finds the matching contract keypair and decrypts the signature of the new contract certificate as described in Section 5.6.2, it can use it in accordance with Section 5.6.2.

### 5.6.4. Evaluation

In order to evaluate whether our new security extension HIP 2.0 meets the declared design goals, we analyze how the proposed solution addresses the security and functional requirements defined in Section 5.3.

**Informal Security Analysis.** The security of the HIP 2.0 security architecture relies on the security of the vehicle's HSM as the trust anchor, as we already discussed with regard to the solutions TrustEV (cf. Section 5.4) and HIP (cf. Section 5.5) that HIP 2.0 builds on. To provide the necessary security properties, this HSM needs to comply with TPM 2.0 specifications from TCG and use CC EAL4+ certified hardware qualified for automotive applications. HIP 2.0 adapts the previously introduced HIP architecture to the distributed and asynchronous provisioning strategy, where EVs can generate own keypairs for their contract credentials and make them available to eMSPs for the out-of-band provisioning in according to VDE-AR 2802 [78]. In the following, we discuss how HIP 2.0 addresses the security requirements defined in Section 5.3.1.

$RS_1$ *Secure key storage:* Assuming the security of the deployed TPM 2.0, all keys that are stored under TPM's protection are secure against powerful adversaries. As described in Section 5.6.1, all PnC keys and SRK also carry the attributes *fixedTPM* and *fixedParent* to prevent the export of their private keys. Backend systems do not store private keys, except for ephemeral keys used only once. Thus, neither a backend hacker nor a car hacker can read out private keys of PnC credentials, and security requirement $RS_1$ is fulfilled.

$RS_2$ *Secure cryptographic operations:* The security of the cryptographic operations is also ensured by the TPM 2.0, which is the only component that can use private keys of the PnC credentials. Since the critical private keys are only used in the shielded area of the TPM and cannot be exported, the requirement $RS_2$ is also fulfilled.

$RS_3$ *Key usage authorization:* All PnC keys are generated with a default access policy, which the EVCC's OEM can use to seal these keys to suitable conditions, e.g., a firmware state or a password. Since such policy (or a combination of policies) needs to be validated before loading the key and the access to this key is not possible if the validation fails, the requirement $RS_3$ is met.

$RS_4$ *Secure key provisioning:* Since all PnC keys are generated directly in the vehicle's TPM and cannot be exported, this requirements is also fulfilled.

$RS_5$ *Cryptographic agility:* The TPM 2.0 specifies a cryptographically agile interface, which allows for the exchange of key lengths and algorithms. However, this ability is somewhat limited by the TCG algorithm registry [116] and existing products. For example, there is currently no hardware available that supports ISO/FDIS 15118-20. HIP 2.0 does not depend on any particular cryptographic algorithms, therefore, insecure cryptographic algorithms can be easily exchanged as long as the compatibility to ISO 15118 [38, 183], VDE-AR 2802 [78] and TPM [114] is assured. In deployed systems, this would either require updating the firmware of the TPM or, if this is not possible, replacing the TPM. The security requirement $RS_5$ is considered fulfilled.

$RS_6$ *Secure key generation:* The OEM provisioning keypair $PC$ and the contract credential keypair $CC$ are generated by the TPM (using the TPM's random number generator) and stored within the protected TPM key hierarchy (cf. Sections 5.6.2). Assuming the key generation of the deployed TPM is implemented correctly, $RS_6$ is met.

$RS_7$ *Trustworthy credential enrollment:* TPM's credential protection mechanism ensures that the EVCC can decrypt the signature of its new contract certificate and use the contract credential only if the following conditions hold: the corresponding contract key is generated and stored in the same TPM as the SRK and the OEM provisioning key; this TPM key is sealed to the defined policy and its public part has the same attributes as those in the CSR. With HIP 2.0, the EVCC provides the direct (by signing the CSR) and indirect (through credential activation) proof-of-possession of the contract private key. Therefore, the security requirement $RS_7$ is fulfilled, too.

**Analysis of Functional Requirements.** For the functional evaluation, we use our TrustEV test-bed setup described in Section 5.4.4 enhanced with the HIP 2.0 functionality. Thus, we modify the software of the EVCC and the SECC components, whereby the SECC hardware also hosts the backend systems of eMSP, CPS and GCS. We use an automotive qualified and Common Criteria EAL4+ certified Infineon OPTIGA TPM SLI 9670 [191] for performance measurements, thus, allowing for realistic benchmarks even with the controllers implemented using general-purpose hardware. All measurements and baselines refer to the DIS version[19] of ISO 15118-20 [183] and are also valid for the previous edition ISO 15118-2 [38] with the exception for certificate updates that are not supported anymore in the newer version. This results are not valid for ISO 15118-20 FDIS due to extensive changes in the security concept including different cipher suits as discussed in Section 5.4.6. We summarize our analysis results with respect to the functional requirements below.

$RF_1$ *Minimal overhead:* HIP 2.0 incurs overhead in the credential provisioning process of ISO/DIS 15118-20 to enable support of VDE-AR 2802 [78]. To show that $RF_1$ is fulfilled, we verify that this overhead does not exceed the standard's limits. With respect to the communication overhead, the OEM provisioning certificate results in 676 bytes (increase of 171 bytes due to the new extension). The contract certificate with an encrypted signature was 538 bytes (increase of 32 bytes due to IV and HMAC tag). Therefore, both certificates stay within the 800 bytes limit of ISO 15118-2[38] and ISO/DIS 15118-20 [183]. As also mentioned before, the fulfillment of this restriction is only possible if the SA does not use the complete allowed size to provide own data. For HIP 2.0, the encrypted session key $SK_{ccert}$ is transmitted in the contract key field of $CertInstallRes$ and requires 52 bytes with SHA256, staying within the ISO/DIS 15118-20 limit of 64 bytes. The CSR of 198 bytes requires a new field and thus is not addressed in the standard's XML Schema Definition (XSD), i.e., requires changes in the data structure. The total communication overhead is with 369 bytes in certificate installation requests and 18 bytes in the respective responses relatively low.

Regarding computational overhead, the changes are only related to the eMSP and the EVCC. The rest of the systems are expected to ignore the introduced changes. We do not consider changes in the GCS because associated timeouts

[19] https://www.iso.org/stage-codes.html

are not defined. Since VDE-AR 2802 allows the eMSP to perform the provisioning processes in advance, additional operations do not affect the ISO 15118 protocol flow. Thus, we focus on the overheads at the EVCC due to the added TPM interactions for key generation, key usage for signatures including policy assertions, and decryption of the contract certificate's signature using command `TPM2_ActivateCredential`. We assume that the difference between the additional symmetric decryption of the contract certificate's signature in HIP 2.0 and the symmetric decryption of private contract key in ISO/DIS 15118-20 is negligible and only analyze the difference of performing the *ActivateCredential* operation and the usual ECDH protocol, both resulting in a symmetric key required for the decryption. The respective times were measured in isolation, i.e., without ISO/DIS 15118-20 communication. All measurements were repeated 100 times resulting in the following averages: 216.4 ms for the generation of contract keys, 552.1 ms for signatures with `TPM2_PolicyPCR` and 230.9 ms for the `TPM2_ActivateCredential`.

In order to optimize this PnC service delay, some operations can be performed in advance, before the PnC session starts. This applies to the contract key generation, and the signing of $CertInstallReq$ message body and CSR, because these operations do not require any session-specific information. Basically, these operations can be performed even during EV preparation. Additionally, [2] shows that it is possible to pre-calculate all TPM policies for time-critical signatures resulting in signature generation delay of 228.8 ms. With these optimizations, the computational overhead of HIP 2.0 is limited to $t_{auth} = 228.8ms - t_{sig}$ during EVCC authentication, where $t_{sig}$ is the time for signature generation, and $t_{dec} = 230.9ms - t_{ecdh}$ during the credential provisioning process, with $t_{ecdh}$ being the time for ECDH-based key generation on the EVCC. Assuming the performance times of an automotive controller as reported in [196][20] equal to $t_{sig} = 46.3ms$ and $t_{ecdh} = 59.8ms$, the resulting computational overhead of the HIP 2.0 security functionality is $t_{auth} = 182.5ms$ and $t_{dec} = 171.1ms$.

$RF_2$ *Conformance to the ISO 15118 protocol specifications:* HIP 2.0 alters the ISO/DIS 15118-20 protocol to increase security. The installation request message is changed, and a minor protocol version number is introduced. We also redefined the fields in the response to send newly defined parameters required by credential protection protocol of the TPM 2.0. The message flow remained unchanged but uses a different credential generation logic, with the EVCC being responsible for contract key generation and certificate requests. In order to integrate these new functionality into the standard, the definition of the respective types is required to preserve the strongly typed XML format. Moreover, intermediate actors may validate the conformance to the message format and discard wrongly formatted messages. In this case, the EVCC may use the received error code to opt for a different provisioning strategy, e.g., for TrustEV. As these changes are required to substantially increase the security, we argue that requirement $RF_2$ is met.

$RF_3$ *Backwards compatibility:* Since HIP 2.0 does not change the general ISO 15118 message flow and the introduced certificate extension is marked as non-critical, the V2G actors that do not wish to support this extension, can simply ignore it. In general case, intermediate nodes (e.g., the SECC) should simply forward the messages without validating their format or drop them causing EVCC to use the default method. Thus, requirement $RF_3$ is fulfilled.

$RF_4$ *Conformance to VDE-AR 2802:* HIP 2.0 requires certain changes for integrating the TPM that influence the functionality of GCS and the pools but the general certificate installation process described in the application guideline remains. Thus, we argue that requirement $RF_4$ is fulfilled.

$RF_5$ *Conformance to standard procedures for CAs:* HIP 2.0 integrates CSRs into ISO/DIS 15118-20 certificate installation messages and makes these requests compatible with the standard format used by CAs for issuing certificates. HIP 2.0 adds an additional step of credential activation to these procedures ensuring that the issued credential can only be used by the designated vehicle. Thus, HIP 2.0 provides stronger security guarantees compared to the common practice. We consider $RF_5$ fulfilled.

### 5.6.5. HIP 2.0 Summary

In this section, we described a new security extension for Plug-and-Charge that strongly reduces the risk of account breaches for e-mobility users to adversaries targeting vehicles or backend systems. The proposed solution HIP 2.0 moves the generation of sensitive contract credentials out of the backend systems of the service providers to the vehicle's TPM and defines a credential activation process compatible with the ISO/DIS 15118-20 protocol flow, application guideline VDE-AR 2802 and standard procedures of the involved CAs. In contrast to these existing credential provisioning methods,

[20] The measurements used a 32 bit Infineon TC297 with the wolfCrypt library and the secp256r1 curve.

HIP 2.0 offers the eMSP strong security guarantees that the credential is indeed used by the intended EV based on the certified features of the TPM 2.0. Our evaluation shows that HIP 2.0, just like HIP, significantly increases the security of PnC and also can be integrated into existing processes and infrastructures in a compliant and backward compatible way. At the moment of publication of [8], only the DIS version of ISO 15118-20 was available, thus, all process descriptions and evaluations with the hardware TPM 2.0 refer to this version. In the upcoming FDIS version, the parts related to the used cryptography undergone a major rework. Since the relevant processes remain valid and HIP 2.0 itself does not depend on any selected cipher suits, the proposed approach is also applicable to the FDIS. The respective proposal for change enabling HIP 2.0 support in the upcoming edition of ISO/FDIS 15118-20 [39] was presented to the Security Task Force of the standardization committee ISO/TC 22/SC 31/JWG 1 responsible for the development of the ISO 15118 standard series. This proposal was postponed to the next revision round of -20, partly because the eMSP representatives actively participating in the committee favored the provisioning strategy, where they retained control over the generation of contract credentials and did not depend on the input from EVs. Also due to time-related reasons, the committee considered the amount of the required changes to the specification too extensive for the present development stage of the standard.

## 5.7. Related Work

Automotive HSMs such as the SHE module [178] or the EVITA HSMs[21] are often implemented as System on Chip (SoC). Based on the results of the EVITA project, several suppliers developed proprietary automotive micro-controllers with integrated HSMs for securing the in-vehicle communication, e.g., for Secure Onboard Communication (SecOC) on the CAN bus [197]. These HSMs provide some basic protection against illegitimate read-out of data but usually do not provide a specially shielded area for secure storage with protection against more sophisticated side-channel attacks. Moreover, HSM implementations, such as SHE [178], are unsuited for the considered use case because they do not support asymmetric cryptography. Some SHE+ extensions are being worked on that introduce asymmetric cryptography to SHE-based HSMs. These are still restricted to the SHE authorizations scheme. The HSM for Vehicle-to-Everything (V2X) communication specified by the CAR 2 CAR Communication Consortium can be either realized as an SoC or as a dedicated chip [181]. The major drawback of these solutions is the limited protection against an attacker with physical presence. This is an advantage of the TPM, which provides, when implemented as a dedicated hardware chip, a specially shielded area for secure storage with a high security level. Most TPM chips have a security certification, e.g., CC EAL4+, and some are also qualified according to the automotive AEC-Q100 standard. Additionally, TPM 2.0 Enhanced Authorization allows for powerful and fine-grained access control.

TPM applications in (electric) vehicles also enjoy a revived interest, partially, due to the enhanced capabilities of TPM 2.0, the availability of open-source software and better affordability of the hardware. So, TPM and Direct Anonymous Attestation (DAA) techniques are the main building blocks of the privacy-preserving solutions in [198, 4] that aim to prevent the generation of movement profiles for drivers based on their authentication data. In [199], the authors propose a TPM-based remote attestation protocol for identity and integrity validation in decentralized V2X networks. Privacy-aware security architectures for V2G networks are presented in [200] and [201]. As part of these solutions, a TPM is used in EV batteries to enable the encrypted communication of charging status, accumulation of information in sealed storage and remote attestation. In [202], the use of a Mobile Trusted Module (MTM) [203] for remote attestation in V2G networks is researched. In their system, the EV is required to present its integrity metric directly to the grid server to proof its state is trustworthy. Formal analysis of the enhanced authorization in TPM 2.0 is performed by the authors of [204].

## 5.8. Chapter Summary

In this chapter, we investigated the security of EV identity credentials used to establish trust in the V2G communication standard ISO 15118. We looked into generation, storage and usage of these credentials as well as various strategies for provisioning and enrollment. According to our security analysis, the connection to the electric grid requires the highest LoA (cf. ISO 29115 [187]) and, therefore, the integration of an HSM into the ISO 15118 processes.

We considered the vehicle's lifecycle from its manufacturing and enrollment at an eMSP, to the usage of PnC and other V2G services as specified in ISO 15118 and VDE-AR 2802. We addressed three provisioning strategies that could be chosen by the eMSP and proposed the security architectures TrustEV, HIP and HIP-20 to enable the necessary protection. From the functional point of view, our goal was to provide solutions compliant to the existing standards and backward compatible with the current functionality to allow for their easy adoption.

The TrustEV architecture addresses the usual ISO 15118 provisioning approach, where contract credentials are generated and delivered by eMSPs on request received from the vehicle during a V2G session. HIP treats an alternative provisioning strategy, where eMSPs can outsource critical cryptographic key generation to the vehicles of their customers if these vehicles meet the necessary security requirements. HIP 2.0 is a further development of HIP, where all cryptographic keys not only generated in the vehicles but also the credential provisioning can take place outside of the V2G charging loop, in accordance to the application guideline VDE-AR 2802 and standardized procedures of the involved CAs. Thus, HIP and HIP 2.0 split the provisioning process of ISO 15118 so that the vehicle first needs to generate its contract keypair and only after that the eMSP can issue the respective certificate for the contract credential.

We defined the components for an EVCC, OEM and eMSP that implement the functionality of TrustEV and HIP. In the case of HIP 2.0, the Global Certificate Store (GCS) also received a new security component. All three proposed architectures rely on an HSM installed in the vehicle or its EVCC for secure generation and storage of cryptographic keys and cryptographic operations. The HSM also provides additional security functionality, such as secure key hierarchy with predefined key attributes, key access policies, direct import of specially encrypted and formatted keys, and a credential

[21] https://www.evita-project.org/

protection mechanism binding credential usage to a particular platform. We used the open standard TPM 2.0 [114] as an instantiation of such HSM. We defined the key hierarchy to store cryptographic keys of V2G credentials, protected by the Storage Root Key (SRK) that is used as an alternative provisioning identity and can never be extracted from the vehicle's TPM. This hierarchy should be created by the vehicle's OEM in the trusted environment, before the vehicle is delivered to its user. In order to transfer this new identity and the associated security parameters from the OEM to the eMSP in an authenticated way that binds the TPM to the vehicle, we specified an optional X.509v3 certificate extension for the vehicle's identity credentials (i.e., OEM provisioning and contract certificate). We integrated the new security functionality into the ISO 15118 protocol flow and described the necessary extensions.

With TrustEV, the contract keys generated by the eMSP are directly encrypted for the TPM import. Since with HIP and HIP 2.0 the EV takes over the key generation from the eMSP, it needs to prove that the keys are generated and stored secure in the vehicle that claims to possess the credential using these keys. This was achieved by implementing the credential protection protocol as part of the enrollment process. This guarantee cannot be achieved with the standard ISO 15118 approach.

All three architectures ensure backward compatibility, i.e., V2G actors, which do not support the respective extension can ignore it and use the legacy approach instead.

In order to evaluate our protocol extensions under realistic conditions, a PoC was implemented using a TPM 2.0 chip available on the market qualified for automotive domain. This choice has multiple advantages. For the companies like Volkswagen [132] that plan to deploy TPMs, our evaluations bring an important insight into potential performance bottlenecks with regard to the V2G scenarios. Moreover, with the extensions being integrated into the new edition of ISO 15118-20, their solutions will be standard-conform[22]. Those companies that opt for an HSM can use our results as a benchmark, because their custom HSMs will have to provide the same security assurance about the management of cryptographic keys, access control and credential protection as we described in this chapter using the TPM interface. This way, they can focus on the particular features, they see necessary considering other security controls in place, and possibly implement them more efficiently. TPMs are known to be relatively slow. Thus, our benchmark describes a very good higher bound for the cost of strong security.

With all three proposed designs, the performance overhead was acceptable for the use case and within the timing constraints of ISO 15118-2. We had to increase the sizes of some message fields that carry additional information, and for HIP and HIP 2.0 even redefine or even add new fields to ISO 15118. This may lead to incompatibilities if intermediate actors validate the size or format of the message before forwarding it.

ISO standards pass through several development stages before they reach the status of the published International Standard (IS). We focus on the Part 2 of the ISO 15118 standard describing the V2G transfer protocol (transport and application layer) and consider several versions in our work. ISO 15118-2 [38] has the status of the international standard. Due to substantial changes required to support emerging use cases such as wireless power transfer and automated connection devices, a new edition ISO 15118-20 for this Part 2 was started. At the moment of the publication of our protocol extensions, the DIS version [183] was available. Usually, after 12 months this version changes its status to FDIS, which is then published by ISO as an official international standard [205]. ISO/DIS 15118-20 has been substantially reworked for its FDIS, including the cipher suits as well as timing and message size constraints, which is unusual. Since the relevant processes remained valid and our security architectures do not depend on any selected cipher suits, they are still compatible with the upcoming FDIS. However, the measurements provided in this chapter do not apply to this latest edition. We plan this as future work when the standard receives the necessary hardware support.

We presented our security extensions to ISO/TC 22/SC 31/JWG 1 "Joint ISO/TC 22/SC 31 – IEC/TC 69 WG: Vehicle to grid communication interface (V2G CI)" responsible for the development of the ISO 15118 standard series. Our proposal for change describing the TrustEV extensions to the protocol successfully passed the committee voting and the TrustEV solution has been officially integrated into the upcoming edition ISO/FDIS 15118-20 (see [39], Chapter 7.9.2.5) with the expected release date in Q2 2022. HIP and HIP 2.0 were not accepted due to major changes in the provisioning logic, and their consideration was postponed to the next revision.

The detailed summaries for the individual security architectures are provided in Section 5.4.7 for TrustEV, in Section 5.5.6 for HIP, and in Section 5.6.5 for HIP 2.0.

Since the EVCC with trust anchor can be removed from the vehicle, i.e., during repairing or scrapping, but still contain sensitive keys inside that can be used by an adversary, in the next chapter we investigate the possibility to bind their usage to the hardware configuration of the complete Electric Vehicle Charging System (EVCS).

[22] Of course, in order to make it happen, Infineon and other TPM manufacturers need to integrate the support of the new ISO 15118-20 cipher suits into the TPM algorithm registry and provide respective hardware.

# 6. Securing Safety-critical Electric Vehicle Charging Systems

In this chapter, we investigate safety and security co-engineering aspects for Vehicle-to-Grid (V2G) systems, where malicious attacks may have both security and safety impacts. Of all domain-specific standards, only SAE J3061 [206] analyzes an integrated safety and security lifecycle for vehicle systems, while ISO 26262 [207] and ISO/SAE 21434 [208] still treat them in an isolated way. We build on the approach for integrated identification of safety and security trust boundaries from [209, 210, 211] and propose a method to enforce the boundary of the safe system operation by binding it to the policy validation within the associated security boundary. The results of this chapter have been published in the paper "Securing Electric Vehicle Charging Systems through Component Binding" [2].

In V2G scenarios, Electric Vehicle (EV) batteries serve as flexible distributed energy resources that help stabilize power supply through managed (dis)charging. The support for bidirectional power transfer is provided by the vehicle's Electric Vehicle Charging System (EVCS) with two connected components: an Electric Vehicle Communication Controller (EVCC) and a Battery Management System (BMS) responsible for the V2G session handling and battery management, respectively. We define the trust boundary for the safety functionality of EVCS as a manufacturer-approved combination of these components and argue that the effective and safe grid integration is possible only when the EVCS is counterfeit-free and protected against malicious attacks. Since Li-ion batteries are prone to overheating and may self-ignite due to improper control [42, 212], with their degradation rate affected by the charging strategy as well [213], EVCS manipulation can cause financial and physical damage, and increase the risk of hazardous situations such as fire and traffic accidents. The possibility to cause safety hazards and the V2G connectivity can provide a strong incentive for malicious attacks aiming to subvert the functioning of the EVCS. For example, if an adversary manipulates the BMS part of the EVCS or replaces it with a tampered one, the battery can be damaged by deliberately operating it outside of the safe range, which can eventually lead to its failure and the danger of fire or explosion [43, 214]. In case several EVCSs connected to the grid are under adversarial control, they can be turned into a botnet of high-wattage devices for a coordinated attack aiming for local power outages or large-scale blackouts [46]. Further risks can originate from the usage of counterfeit components in EVCSs. The growing market and high price of EV batteries attract criminals selling expired or low-quality counterfeit spare parts, which do not meet regulatory standards and are potentially unsafe [215].

**Contributions.** Following SAE J3061, we identify security threats for the EVCS that can cause safety issues. To counter these threats and prevent harmful situations, we propose a novel security architecture `secEVCS`, which guarantees that a vehicle participating in V2G services has a manufacturer-approved configuration of the EVCC and the BMS by securely binding these components, i.e., only an original charging system can (dis)charge electric energy at a charge point.

`secEVCS` uses as security anchors a Trusted Platform Module (TPM) [114] (as vehicle's Hardware Security Module (HSM)) in the EVCC and the Device Identifier Composition Engine (DICE) [216] in the BMS. The general idea is to allow access to a V2G authentication key required for connecting to the grid only if the binding (i.e., security policy) is successfully verified using the TPM's enhanced authorization functionality. In addition, `secEVCS` protects against the installation of counterfeit spare parts and reuse of secrets from scrapped charging system components.

`secEVCS` was implemented using a real hardware TPM 2.0 and ISO 15118 [38, 183] for V2G communication[1] to evaluate the proposed solution under realistic constraints. To the best of our knowledge, TPMs have not been deployed in this scenario yet and the analysis of the associated trade-offs is missing. Therefore, our work aims to close this gap.

This chapter is structured as follows: In Section 6.1, we describe our system model and in Section 6.2 analyze the corresponding safety-related security threats. Security and functional requirements are defined in Section 6.3. In Section 6.4, we introduce our security architecture `secEVCS` before we describe and evaluate our prototype in Section 6.5. We discuss the applicability of our solution in Section 6.6 and provide the related work in Section 6.7. Section 6.8 concludes this chapter.

[1] The ISO 15118 standard series is actively adopted by the industry, e.g., the CharIn network (`www.charinev.org`). While we focus on the current protocol specification, ISO 15118-2, we consider the upcoming edition draft, ISO/DIS 15118-20 [183], whenever relevant.

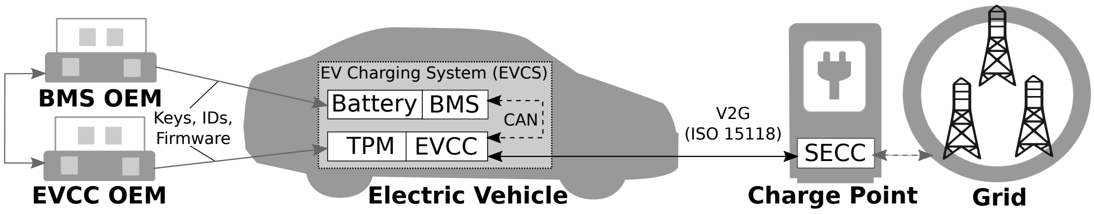


Figure 6.1: System Overview

## 6.1. System Overview

An Electric Vehicle Charging System (EVCS) comprises a Battery Management System (BMS) with an integrated rechargeable battery and an Electric Vehicle Communication Controller (EVCC) with an HSM that provides it with security services. Since openly specified TPMs become more common in modern cars [132], we assume it to implement the HSM functionality, without limiting our system model. Figure 6.1 gives an overview of our considered system.

The BMS's major function is to maintain the vehicle's battery within its safe operating range, to monitor its state (i.e., state-of-charge, state-of-health, and state-of-function) and to assess the available energy amount [217]. The BMS also controls battery cooling/heating, operates power switches, and exchanges charging-related data via a Controller Area Network (CAN) bus with the EVCC. The EVCC is responsible for communication with a Supply Equipment Communication Controller (SECC) of a charge point during V2G service sessions and supports automated authentication of EVs (i.e., Plug&Charge). We assume an EV has to identify and authenticate itself against an SECC by means of a so-called asymmetric *authentication key* stored in its EVCC before it can connect to the grid for charging. In ISO 15118 [38, 39], this key is part of the vehicle's OEM provisioning or contract certificate. When negotiating a charging schedule for a V2G session with the SECC, the EVCC queries the BMS on such parameters as battery state, allowed current and voltage. Together with the expected departure time and other user-defined charging preferences, this information is crucial for demand-side management aimed to improve efficiency and stability of the grid. Grid-friendly charging behavior can be awarded with reduced tariff rates. While charging, the EVCC periodically receives metering receipts from the SECC for signing that may later be used to bill the EV's driver for the charged energy.

The life-cycle of an EVCS and its components includes several stages. The BMS and the EVCC are produced by respective Original Equipment Manufacturers (OEMs) that provide firmware and cryptographic keys. An automotive OEM creates for an EV a unique Vehicle Identification Number (VIN)[2] and authentication key, while a battery OEM provides a BMS with a unique *identity key*. The cryptographic keys are assumed to be created by OEMs in a secure way and not leaked during manufacturing. When deploying a new EVCS in a vehicle, the EV's manufacturer defines a safety-approved configuration, by binding a BMS to an EVCC. Replacing or updating any of the EVCS's components can be carried out in an authorized repair shop, where a new approved configuration will be created.

A drained EV's battery can also be replaced with a fully-charged one in a battery swap station operated by a battery swapping company [218]. We assume backend systems of the OEMs and service providers are able to exchange information securely using a common Public Key Infrastructure (PKI).

## 6.2. Safety-related Security Threats

Vehicle's charging system is a safety-critical system. The growing number of reports on self-ignition of EV batteries while charging or in driving [42], shows the potential for adversaries not only to damage EVs and their components but to harm their passengers or people in the vicinity with targeted attacks. In [214, 219], the authors propose a Security-Aware Hazard and Risk Analysis Method (SAHARA) and use it to identify threats for a BMS and estimate the risk. Threats for BMS and potential effects are also analyzed in [43]. Based on these analyses, we consider the following threat scenarios with their possible safety impact:

**Configuration Tampering.** An adversary replaces the BMS in the EVCS with one that is not approved by the OEM and/or under full control of the adversary. This could also be done by the EV's driver who aims to extend the range of the vehicle by upgrading the battery [220]. Such action would violate the OEM's warranty.

[2] VINs mainly conform to two international standards ISO 3779 and US Standard FMVSS 115; a VIN is always 17 characters long.

**Impact:** The attack affects the EVCS's integrity and has multiple safety implications. The adversary can modify battery information, e.g., to indicate a larger capacity than given, and control battery functions to, e.g., ignore dangerous operating conditions like overheating in order to damage the battery or cause a fire [213]. Moreover, incompatible EVCS components can incorrectly interpret exchanged data, which can shorten the battery lifetime and lead to hazardous situations. Disrupting demand-side management would also affect the grid.

**Charging Contract Hijacking.** An adversary uses a scrapped EVCC that stores the authentication key of a valid V2G user to charge own vehicle on the user's (or, possibly, uncovered) account or to use the access profile to connect to the V2G service. If the adversary is able to extract the key or establish a remote authentication interface using this components, this attack can easily be scaled to multiple vehicles.

**Impact:** The attack affects the confidentiality of the EV's key and the privacy of the previous user of the scrapped EVCC; the integrity and authenticity of V2G sessions is also breached. The latter can affect grid stability due to unexpected charging behavior or even cause blackouts if the attack is launched in a coordinated manner [46].

**Counterfeit BMS.** An adversary uses old BMSs with expired or malfunctioning batteries to produce and sell counterfeit products, which can still carry the label of the original manufacturer but are not certified for safe use.

**Impact:** The attack affects system integrity and authenticity. Counterfeit batteries often lack required safety protections and can easily catch fire.

## 6.3. Security and Functional Requirements

In order to prevent the threats defined in Section 6.2 with `secEVCS`, we propose to enable access to an EV's authentication key needed to use V2G services, only if its EVCS is original, i.e., only if a verifiable binding between the EVCC and the BMS exists. This leads to the following security requirements, which must be fulfilled:

$SR_1$ *Secure private key storage and usage.* Private keys (e.g., authentication keys, identity keys) shall be protected against leakage during their storage and usage.

$SR_2$ *Restriction of key usage to trustworthy systems (Key usage authorization).* Access to private keys shall be possible only if the EVCS is trustworthy, i.e., the components configurations are approved by the manufacturer and are not manipulated.

$SR_3$ *Revocation support.* It shall be possible to revoke BMS of an EVCS in case it is removed from an EV, so that it cannot be used in another manufacturer-approved EVCS configuration later on.

A security solution for EVCSs should bring clear benefits to the automotive and EV battery industry and consumers without unnecessary restricting legitimate usage scenarios. This results in the following functional requirements:

$FR_1$ *Minimal performance overhead.* A solution shall not cause undesirable delays in EV charging and shall meet timing constraints of standard V2G protocols. In ISO 15118, e.g., charging start may be delayed due to two operations using the EVCC's authentication key and each of these operations is subject to strict timeouts (details in Section 6.5.2).

$FR_2$ *Support of legitimate component exchange.* Only legitimate entities shall be able to replace or swap the battery (including BMS) and/or EVCC while maintaining the manufacturer-approved EVCS configuration.

## 6.4. `secEVCS` Security Architecture

The general idea of `secEVCS` is to bind EVCC and BMS of an EV and to allow access to an authentication key only if this binding can be verified. The authentication key is securely stored and used in the EVCC's TPM and access is only possible if a TPM enhanced authorization policy is fulfilled. This policy includes the result of challenge-response protocol between EVCC and BMS.

`secEVCS` consists of an initial EVCS preparation phase for initializing and binding EVCC and BMS (cf. Section 6.4.1), the EVCS usage phase (during the lifetime of the EV) supporting the authorization of charging sessions and the swapping of batteries (cf. Section 6.4.2), and performance optimizations (cf. Section 6.4.3). Figure 6.2 shows the enhanced authorization policy verification steps as the central part of `secEVCS`, which are described in more detail below.

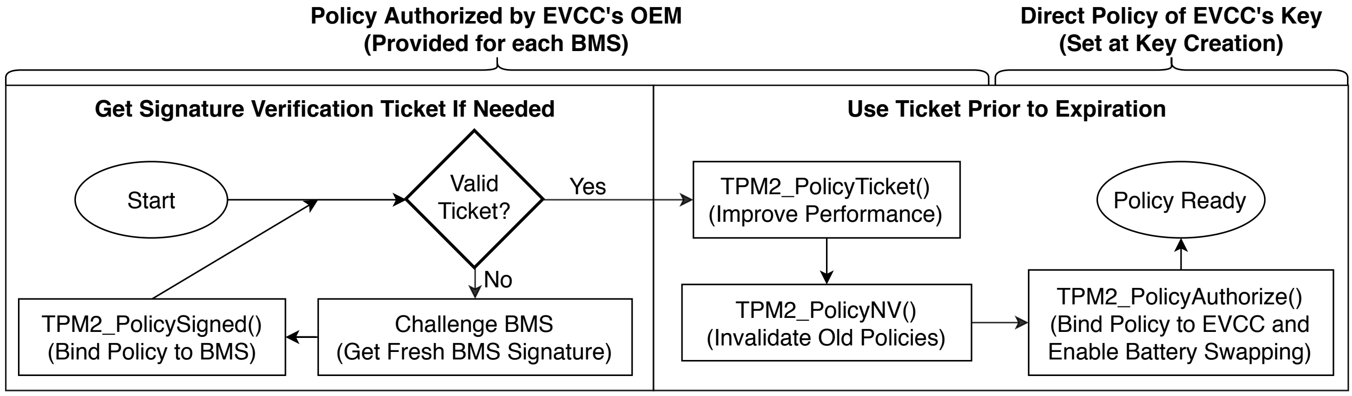

Policy Authorized by EVCC's OEM
(Provided for each BMS)
Direct Policy of EVCC's Key
(Set at Key Creation)
Get Signature Verification Ticket If Needed
Use Ticket Prior to Expiration
Start
Valid Ticket?
Yes
No
TPM2_PolicySigned()
(Bind Policy to BMS)
Challenge BMS
(Get Fresh BMS Signature)
TPM2_PolicyTicket()
(Improve Performance)
TPM2_PolicyNV()
(Invalidate Old Policies)
TPM2_PolicyAuthorize()
(Bind Policy to EVCC and Enable Battery Swapping)
Policy Ready

Figure 6.2: secEVCS Policy Verification Steps

### 6.4.1. EVCS Preparation

**EVCC Preparation.** During manufacturing, the OEM generates an authentication key on the EVCC's TPM. This authentication key comes with an authorization policy (*TPM2_PolicyAuthorize()*) that refers to an OEM public key (cf. Figure 6.2 on the right) for policy statements. To use the authentication key, the EVCC software needs to present a policy to the TPM that was authorized (signed) by the OEM. Thus, the newly created key cannot be accessed directly after its generation. The OEM needs to explicitly issue (and sign) a policy statement that describes, under which conditions the authentication key can be used. We use the EV's VIN as policy reference value of the authentication key's policy. This way, this key can only be accessed if a policy is fulfilled that was authorized by the OEM with the corresponding VIN denoted during key generation, i.e., a signed policy addresses the intended EVCC only and cannot be copied to other EVCCs. The EVCC preparation process is represented by InitTPM() in Figure 6.3. If key generation on the TPM is not possible, keys can be also generated outside and imported into the EVCC's TPM (e.g., in ISO 15118 using TrustEV [1]).

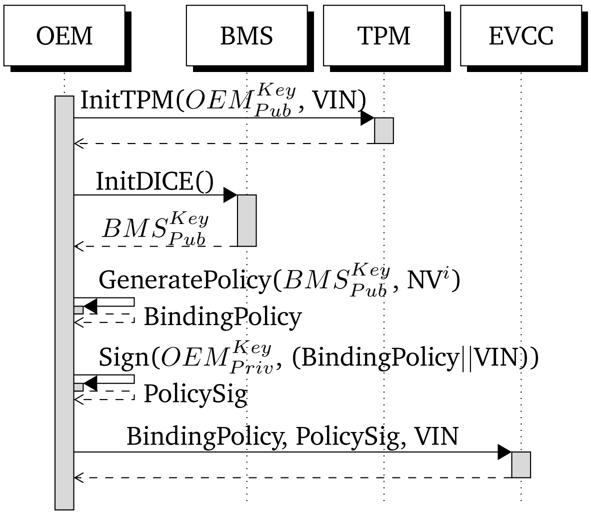

OEM
BMS
TPM
EVCC
InitTPM($OEM_{Pub}^{Key}$, VIN)
InitDICE()
$BMS_{Pub}^{Key}$
GeneratePolicy($BMS_{Pub}^{Key}$, NV$^i$)
BindingPolicy
Sign($OEM_{Priv}^{Key}$, (BindingPolicy||VIN))
PolicySig
BindingPolicy, PolicySig, VIN

Figure 6.3: EVCS Preparation

**BMS Preparation.** The BMS is equipped with a DICE [216] (cf. InitDICE() in Figure 6.3), as a cheap alternative to a TPM suitable for highly constrained embedded systems [221]. DICE generates a unique device Identifier (ID) based on a globally unique secret and a measurement of the device's first mutable code using a cryptographically secure one-way function. Hence, any persistent attack to the BMS results in the generation of a different device ID. As the DICE is trusted and has exclusive access to the unique secret, it is impossible for an adversary to recover the secret or generate a valid device ID after a persistent attack. The BMS can use the DICE-generated ID to secure its identity key (e.g., by using the ID as seed to a Key Derivation Function (KDF) and using the resulting key to encrypt the identity key before it is stored). This way, the BMS's identity key is also protected from persistent attacks. With this key, the BMS can authenticate itself using a public key signature. As the BMS' public identity key is required for the binding between EVCC and BMS (see next paragraph), the key is read out by the BMS' OEM and passed to the EVCC's OEM (cf. OEM in Figure 6.3).

**BMS and EVCC Binding.** At this step, the EVCC's OEM needs to issue (sign) a respective policy (cf. BindingPolicy and PolicySig in Figure 6.3). The policy consists of a *TPM2_PolicySigned()* containing the public key of the BMS. To fulfill this condition, a nonce generated by the TPM must be signed with the BMS' private key and the signature validated by the TPM. Additionally, the policy contains a *TPM2_PolicyNV()* statement that links this policy to a monotonic counter inside the TPM. If a BMS binding needs to be revoked in the future, the OEM can increment the TPM's counter. The signature over this policy by the OEM also includes the VIN as policy reference value as mentioned above. This binding can happen in conjunction with the initial key generation or at a later stage.

### 6.4.2. EVCS Usage

**Charging Authorization.** Access to the authentication key is only possible if the BMS and EVCC binding is successfully verified. This process is shown in Figure 6.4 (PolicyCheck() aggregates all policy-related validations). The EVCC first loads the authorized policy (i.e., the BMS binding policy) and policy reference value, i.e., VIN, and verifies the signature using the OEM's public key. The result is a so-called signature validation ticket. Then, the EVCC starts a policy session (*TPM2_StartAuthSession()*) and sends the session's nonce as a challenge to the BMS. The BMS signs the nonce with its private key and returns the signature and its public key. The EVCC extends its session with a validation of the BMS' signature (*TPM2_PolicySigned()*) and the comparison of the Non-Volatile (NV)-counter (*TPM2_PolicyNV()*). The BMS binding is authorized using the signature validation ticket (*TPM2_PolicyAuthorize()*). After this, the policy session is in a state that grants access to the EVCC's key operations and the EVCC can issue a *TPM2_Sign()* operation to authenticate itself against the charge point.

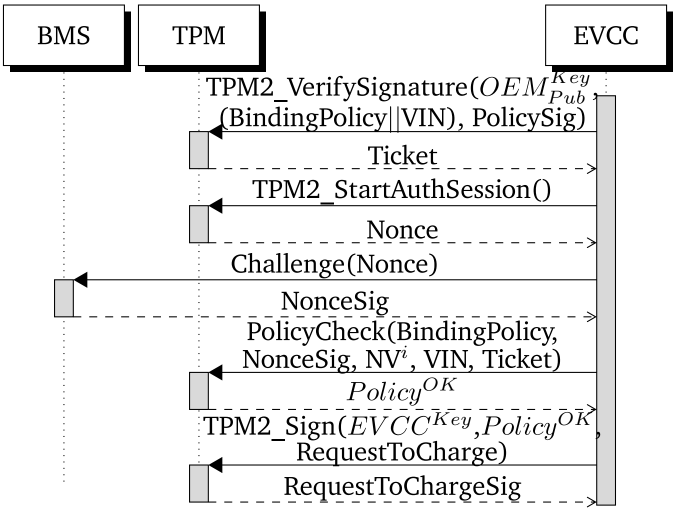

BMS
TPM
EVCC
TPM2_VerifySignature($OEM_{Pub}^{Key}$,
(BindingPolicy||VIN), PolicySig)
Ticket
TPM2_StartAuthSession()
Nonce
Challenge(Nonce)
NonceSig
PolicyCheck(BindingPolicy,
NonceSig, NV$^i$, VIN, Ticket)
$Policy^{OK}$
TPM2_Sign($EVCC^{Key}$,$Policy^{OK}$,
RequestToCharge)
RequestToChargeSig

Figure 6.4: EVCS Usage

**Battery Swapping.** A battery swapping company needs to maintain a backend connection to the OEMs and perform the above BMS binding process. To invalidate the binding to the old BMS, the OEM increments the TPM's counter and then issues (signs) a new policy for the new BMS and the new counter value.

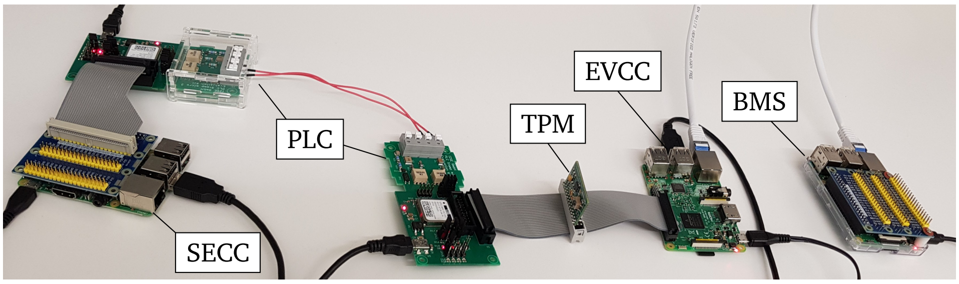


Figure 6.5: Test-bed Setup

### 6.4.3. EVCS Enhancements for Better Performance

The process for key usage described in Section 6.4.2 requires the EVCC to challenge the BMS and perform the policy session assertions for each access to the authentication key. This can lead to undesirable delays when trying to charge an EV (e.g., in our tests it took on average 2.4 seconds; cf. Section 6.5.2). This can be easily avoided by sending challenges to the BMS independent of the charging sessions and pre-calculating the entire policy session. For example, the EVCC can send a challenge whenever the charging port lid is opened. This way, a correct policy session is always available before the authentication key is to be used.

Another issue is that the EVCC has to send a new challenge to the BMS each time it needs to use the authentication key. This can delay communication protocols between EVCC and SECC using this key not only for charge authorization, but also, e.g., to sign metering receipts. A low performance of the BMS Electronic Control Unit (ECU)[3] and a low throughput of a CAN bus[4], this process (estimated to about 5.8 seconds) may exceed timing constraints of the protocol.

We address this issue by using a shortcut in the *TPM2_PolicySigned()* command. The command can output a ticket upon validation, which can be used in future policy sessions (within the expiration time) using the *TPM2_PolicyTicket()* command as a replacement (cf. Figure 6.2 on the left). This expiration time should not be too short (to gain a speedup) and neither too long (to restrain attacks). An expiration time of 5 minutes is a good starting point to give a user enough time to initiate charging, while still preventing potential attacks. These 5 minutes provide enough time to start a second policy session from the beginning.

## 6.5. Implementation and Evaluation

In this section, we evaluate our proposed solution. We use ISO 15118-2 [38] as communication protocol between EVCC and SECC. We describe the implemented prototype and evaluate the added overhead. A minimal overhead is important for the usability of secEVCS in terms of compliance to the timing constraints of ISO 15118 on EVCC signature creation as well as for user convenience. Additionally, ISO/DIS 15118-20 [183], the upcoming successor of [38], allows for even tighter timing constraints, which are also considered in the evaluation.

### 6.5.1. secEVCS Implementation

Our prototype was implemented using three Raspberry Pi 3 Model B running Linux kernel 4.14 to simulate the EVCC, BMS, and SECC. The EVCC-Pi is equipped with an Infineon Iridium 9670 TPM 2.0. EVCC and BMS communicate over regular Ethernet, while SECC and EVCC communicate over power-line communication (PLC) Stamp micro 2 EVBs (similar to PLC over a charging cable). Our test-bed is shown in Figure 6.5.

To execute TPM commands, we use the TPM2-TSS[5] and as ISO 15118 implementation we use RISE-V2G[6], integrated with the TrustEV implementation from [1] for *EVCC Preparation* (cf. Section 6.4.1). The challenge-response communi-

[3] It can take an ARM Cortex-M0+ without performance optimizations up to 3649 ms to create a signature using the algorithm and parameters defined by ISO 15118 [222].

[4] Transmitting 16 byte nonce, 64 byte EC public key, and 64 byte ECDSA signature in 18 extended CAN frames (16 bytes each with 8 bytes data and 7 bits inter-frame spacing) with 125 kbps Low-Speed CAN takes about 20 ms under optimal conditions.

[5] TPM2-TSS: `https://github.com/tpm2-software/tpm2-tss`

[6] RISE-V2G: `https://github.com/V2GClarity/RISE-V2G`

cation between EVCC and BMS is implemented using the Secure Shell (SSH) protocol [223] to simulate any added security means on the automotive bus.

The expiration time of BMS signatures is set to 5 minutes. Challenges are sent to the BMS 5 seconds before start of ISO 15118 communication (to simulate the time from opening the charging port lid to plugging in the charging cable) and a minute before the current signature expires. After receiving a signature, *TPM2_PolicySigned()* is called to retrieve the verification ticket. The ticket is used by two processes to pre-calculate multiple policy sessions concurrently. When the authentication key is used in ISO 15118, one of these pre-calculated policies is consumed. We only use two pre-calculation processes along with the ticket generation process, to not exceed three concurrent authorization sessions. While a TPM must be able to support 64 active sessions, it must only be able to hold 3 of those in RAM at a time [224]; hence, exceeding this limit would decrease performance on TPMs that only support the minimums from [224].

### 6.5.2. Performance Evaluation

During performance evaluation, we measured the computational overhead created by our prototype from Section 6.5.1 compared to the default RISE-V2G implementation. All measurements were repeated 100 times each using Java's *System.nanoTime()*. During a charging loop, the EVCC alternates between sending charging status and signed metering receipt messages. It tries to send them as fast as possible, reaching 121.9 ms between consecutive receipts. For our measurements, the EVCC sent 10 metering receipts for each of the 100 charging loops.

The time for signing ISO 15118 messages with default RISE-V2G was 15.7 ms and with secEVCS 469.8 ms. For comparison, without the parallel pre-calculated policies the average signature time was 1119.8 ms. Without any of the performance optimizations for secEVCS, i.e., when an on-demand challenge is sent to the BMS for each key usage and no policy pre-calculation is done, the time for signing ISO 15118 messages was 2437.8 ms. Our measurements for secEVCS are shown in Figure 6.6 (signature #0 is for charge authorization and #1-10 for metering receipts).

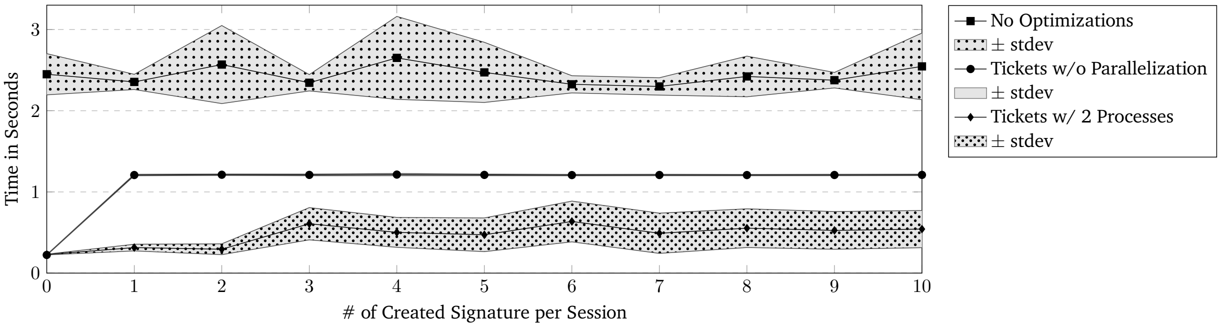


Figure 6.6: Mean Times of Signature Creation in a Charging Session

With our setup, the time from sending a challenge to the BMS until receiving a signature was 277.6 ms. In Section 6.4.3, we discussed a more realistic device configuration. Extrapolating our measurements to low power ECUs and CAN bus, the measurements for BMS signatures would increase to 3669 ms, leading to ISO 15118 message signing time of 5829.28 ms for secEVCS without optimizations. This correlates to the head time used for pre-calculation of sessions and the use of the improvements proposed in Section 6.4.3.

It is worth noting that with secEVCS there was a significant difference in the times for signing charge authorization requests compared to metering receipts. The former was on average 228.8 ms, whereas the latter 493.9 ms. Also, the mean time for signing the first 2 receipts of each charging loop was 304.3 ms and the mean time for the 3rd to 10th was 541.3 ms. The reason is that we have only two processes to pre-calculate policy sessions. Therefore, when starting a charging session, there are two policies ready to use and if more than two signatures need to be created, new policy sessions need to be calculated at run-time. In our setup with 2 parallel policy sessions ($n_p$), 121.9 ms between metering receipts ($t_m$), and the signature time of 228.8 ms ($t_s$) there are only 472.6 ms ($= n_p \times t_m + (n_p - 1) \times t_s$) for policy pre-calculation. With an average time for policy calculation of 737.4 ms, i.e., an overrun of 264.8 ms, this gives about 500 ms for signatures without full policy pre-calculation. While this should lead to alternating signature times (after a signature with $t_s = 500$, the available time for pre-calculation is 743.8 ms which should allow for a fast signature), we did not experience this effect. Instead, as a result of the parallelization, the times for the 3rd metering receipt signature onward were much less predictable, with a standard deviation of 231 ms compared to the times for the first 2 signatures with a standard deviation of 56.4 ms and the time for the authorization signature with a standard deviation of 4.8 ms.

Since in ISO 15118 at most the first two signatures are time-critical, i.e., signing a request for a new authentication key and signing the charging authorization request can delay the V2G session start, we argue that the achieved results are acceptable for the use-case. Regarding ISO 15118 compatibility, the only requirements affected by the increased EVCC signature time are the *V2G_EVCC_Sequence_Performance_Time* of 40 seconds (time for the EVCC to send its next request after a response from the charge point) and the *V2G_SECC_Sequence_Timeout* of 60 seconds (timeout of the charge point for waiting on the next EVCC request). Even without the performance optimizations, `secEVCS` stays well within the relevant limits. However, in the new edition ISO/DIS 15118-20 [183], the timeout mechanism for metering receipts was changed. The SECC may define its own arbitrary timeout in seconds. Therefore, a minimal timeout of 1 second is possible and only the optimized `secEVCS` would be able to meet this minimum (cf. Figure 6.6).

## 6.6. Requirements Coverage Discussion

To prevent safety-related threats from Section 6.2, `secEVCS` verifies the binding between the EVCS components prior to charging. In Section 6.3, we defined the requirements that need to be satisfied by a secure and usable solution. In the following, we informally discuss how these requirements are covered by `secEVCS`.

In `secEVCS`, the EVCC's authentication key is generated and stored in the controller's TPM and can only be accessed by this TPM and used only in its shielded location. Thus, this private key is protected from any attacks that read keys from the EVCC's memory. Since the binding between EVCC and BMS is validated based on a signature by the BMS's identity key, secure storage of this private key is essential for the overall security. To protect this key, `secEVCS` uses DICE, a smaller security architecture with low hardware requirements suitable for resource-limited systems. Due to the relatively high cost of a TPM, it is also desirable to limit their usage to externally facing ECUs. Thus, `secEVCS` meets the security requirement *Secure private key storage and usage ($SR_1$)*.

The TPM always verifies, whether the EVCC is in a trustworthy state and whether the BMS defined in the configuration provided by its OEM is present, before allowing access to the authentication key. Thus, if an adversary has manipulated or replaced the BMS, or uses a scrapped controller, the EVCC will not be able to authenticate itself for using V2G services. This corresponds to the security requirement *Restriction of key usage to trustworthy systems ($SR_2$)*.

The security requirement *Revocation support ($SR_3$)* is fulfilled by validating the value of a monotonic counter inside the EVCC's TPM, which can be incremented each time an expired or malfunctioning BMS is exchanged in a repair shop. This way, it will not be possible to use this BMS together with the EVCC for charging, because it is not part of the approved configuration anymore.

The functional requirement *Minimal performance overhead ($FR_1$)* is met as explained in detail in Section 6.5.2. Requirement *Support of legitimate component exchange ($FR_2$)* is also fulfilled, since only an authorized OEM or service provider can register a new BMS with an EVCC's TPM by sending an updated policy.

## 6.7. Related Work

Automotive security is a topic of active research since a decade. Various local attack vectors were analyzed including tire pressure monitoring systems [225], Bluetooth and cellular radio [226], wireless and a diagnostics port [227], or a malicious app in the infotainment system as in Kia Cee'd [228]. The possibilities of the adversarial control over the vehicle were discussed in [121], and demonstrated in practice by Miller and Valasek in the famous "Jeep hack" [129]. First remote attack vectors were demonstrated by Miller and Valasek in [59] and [229]. With increasing connectivity of the modern cars, more remote attack vectors are discovered. For example, remote attacks on Tesla vehicles were shown in [230], [60], and [61], and the feasibility of remote targeted attack on multiple Internet-Connected BMW vehicles in [62]. These attacks prove that an adversary can get access to basically any ECU and manipulate its operation. All these attacks have, therefore, also a strong safety impact.

The analysis of research related to the usage of HSMs and particularly TPMs to protect vehicle credentials and implement policy mechanisms, was carried out in Section 5.7 and also applies here. To the best of our knowledge, this is the first attempt to address the problem of EVCS's safety using TPM-based component binding.

## 6.8. Chapter Summary

Recent studies identify fire and traffic incidents caused by manipulated EVCSs as a major safety concern for EVs. We propose a new security architecture `secEVCS` aiming to prevent harmful situations by allowing only vehicles with manufacturer-approved charging systems to (dis)charge electric energy at charge points. This guarantee is achieved through securely binding the vehicle's components EVCC and BMS responsible for management of the charging process by using a set of policy assertions that restricts their functionality to a certified configuration. This binding needs to be validated each time the EV wants to use its authentication credential for V2G services, which turned out to be challenging with regard to the user convenience and communication timeouts. In general, this approach may be extended to include further components into this validation presuming that the resulting overhead is acceptable.

In order to evaluate our solution within realistic constraints, we implemented `secEVCS` using the enhanced authorization feature of a TPM 2.0 chip and ISO 15118-2 [38] as a V2G protocol. Also, the draft of the next edition ISO/DIS 15118-20 [183] that defines harder timing constraints was considered in our evaluation. While the performance overhead is acceptable for the use case and within the timing constraints of ISO 15118-2, a straightforward approach of TPM-based component binding cannot meet the new requirements. With the new edition, conformance to the standard can only be guaranteed if all proposed `secEVCS` performance optimizations are in place.

Our results provide a useful reference for future work that can address the shown limitations (e.g., timing conditions or runtime attacks on EVCS) or adopt `secEVCS` as a security anchor in broadened scenarios, e.g., secure load management.

# 7. Securing Railway Command and Control Systems

In this chapter, we investigate safety and security co-engineering for safety-critical railway Command and Control Systems (CCSs) based on digital interlocking. The results of this chapter have been published in the papers: "Security requirements engineering in safety-critical railway signaling networks" [10], "A reference architecture for integrating safety and security applications on railway command and control systems" [11], "Implementing a security architecture for safety-critical railway infrastructure" [9], and "Security requirements for internet of railway things" [24].

The safety-critical railway infrastructure is currently undergoing a digitalization process. The railway CCSs are changing with the use of Commercial of the Shelf (COTS) products and IP-based networks, as well as the overall increasing interconnection between systems' components. Previously used closed and manufacturer-specific solutions are increasingly being replaced by standard hardware and software technologies. This change can be observed in the European Union (EU) and worldwide. Consequently, the digitalization of railway systems is on the agenda of the EULYNX Cluster, an European initiative for standardization of interfaces and elements of signaling systems[1]. In Germany, Deutsche Bahn (DB) realizes plans to digitalize its infrastructure as part of the NeuPro project [63]. The first step is the digitalization of Object Controllers (OCs). Since the DB railway network consists of more than 3,300 Interlocking Systems (ILSs) and more than 200,000 field elements, the integration of Information Technology (IT) into control processes aims to increase efficiency of railway operations.

However, the railway digitalization is also associated with higher risk of malicious attacks, making it imperative to jointly examine safety and security [66, 67]. Integrating security mechanisms into a safety-certified system while providing freedom of interference necessary to keep safety certification for this system is a major challenge [64].

**Contributions.** In this chapter, we propose a security architecture for safety-critical railway infrastructure enabling the joint operation of safety and security mechanisms on a single hardware platform. The architecture consists of a hardware platform with a Trusted Platform Module (TPM) 2.0 as trust anchor, the Multiple Independent Levels of Security (MILS) Separation Kernel (SK), and various security applications.

The chapter is organized as follows. Section 7.1 presents the system model. In Section 7.2, we provide a review of various adversary profiles relevant for the railway CCS. Section 7.3 details the conducted threat analysis using DIN VDE V 0831-104. The elicited security requirements are presented in Section 7.4. Section 7.5 describes and discusses the proposed security architecture. Section 7.6 analyzes how our security architecture fulfills the security requirements. Section 7.7 concludes the chapter.

## 7.1. System Model

Current signaling systems can be divided into three layers: operational, interlocking, and field element layer [64]. In our system model, we consider the interlocking layer and the field element layer of the railway signaling architecture depicted in Figure 7.1. In orange, the safety-critical functionality is shown.

The operational layer (not shown) contains Operations Control Center (OCC) responsible for the supervision of rail operations including the control of train movements. This is done by operators on the specialized workstations, which consist of a safe display of the controlled area. Beside the operators, also Security Operations Center (SOC) systems as well as disposition systems are located on this layer. The buildings where these systems are located have to fulfill special requirements regarding physical security.

Most of the important safety systems are located at the interlocking level. The ILS is responsible for the safe operation of trains, i.e., for determining of technical dependencies for train routes and sending commands to proper field elements. In case of an error or a fault occurring in a field element, the ILS switches to the safe state (fail-safe) and blocks the route until the dependency is restored. The Maintenance and Data Management System (MDM) is in charge of providing software updates for the components in the interlocking and field element layer, logging of diagnostic data and potential

[1] `https://eulynx.eu/`

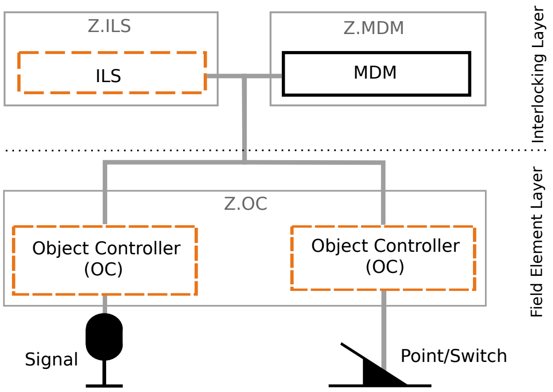


Figure 7.1: General Railway Signaling Architecture based on NeuPro

security events, and time synchronization. In addition, the MDM may forward the security-related messages to the SOC. Components on this layer are developed according to several safety standards like EN 50126 [231] and only the required functionality is available. Additionally, these components are built redundant, which means that in case of a defect one of the standby systems comes in place and the maintenance personnel is notified to replace the failing component. The data networks and the power supplies are redundant, too.

Field elements located on the lower layer are sensors and actuators, such as light signals, level crossings, points/switches, Train Detection System (TDS) or other track-side equipment of this type. An OC usually controls exactly one field element and provides an interface to the ILS by translating digital interlocking commands into electrical signals that steer the field element and by reporting the element's state back to the ILS. As a rule, the OC has no "intelligence" of its own. OCs have limited physical protection provided by the junction boxes in contrast to the components of the interlocking layer located in the buildings with physical access control.

All components down to OCs are connected via the so-called railway Wide Area Network (WAN), i.e., Ethernet- and IP-based communication network owned or leased by the railway operator. Railway operations that mandate resilient transport and reliable message delivery may use the Rail Safe Transport Application (RaSTA) protocol [232].

According to DIN VDE V 0831-104 [66], we split the system model into logical zones, where components are assumed to have similar security requirements. We define three zones: `Z.ILS`, `Z.MDM`, and `Z.OC` (cf. Figure 7.1). Each zone definition includes the list of objects in the zone, the logical and physical borders, the data flows between the zones, the interfaces (Ethernet), and the physical connections.

**Security Goals.** Since the threat landscape of railway CCSs changes, hazardous situations can equally result from random hardware faults or software bugs and be caused by actions of a malicious adversary. Therefore, security goals in addition to safety goals need to be considered. We define the following security goals:

**Availability:** The railway CCS should at any time be able to provide its required functionality and data, i.e., to generate safe routes, send and receive signals and commands over the network, and log critical events. This requires provisions against Denial of Service (DoS) attacks that can be carried out on a network or a cyber-physical layer and block or delay time-critical operations. In safety systems, availability guarantees are usually achieved through redundancy. In case of a motivated adversary, this might not be enough, especially if redundant components "fails-safe" silently, and the attack stays undetected until the system limit is reached.

**Integrity:** Considering that a railway CCS is a highly distributed and complex system, it requires the protection against any unauthorized modification of its data (configurations, commands, access credentials) as well as software and hardware components. If such modifications stay undetected, the correct operation of the CCS can be disrupted in multiple ways.

**Authenticity:** It is necessary to be able to verify the trusted origin of safety- and security-critical data and components in order to prevent, e.g., that tampered software or hardware is deployed in a railway CCS or reactions build on the falsified information.

**Confidentiality:** Data transferred by safety applications in a CCS are not considered confidential. Apart from safety assets, the electronic interlocking system architecture contains security assets such as access credentials or cryptographic keys for the PKI that need to be protected from unauthorized disclosure or use.

**Accountability:** Any action performed by a CCS should be traceable to an authorized entity responsible for this action.

**Non-repudiation:** An authorized entity in a CCS should not be able to deny its actions.

**Auditability:** Security-related events need to be recorded in an auditable form (including time, source, user, etc.).

## 7.2. Analysis of Adversary Profiles

Classification of potential adversaries is an important step of threat analysis and risk assessment. For this task, an adversary profile or an adversary model can be used [233]. An adversary profile defines categories of adversaries giving a general idea about their motivation, skills, resources, likely targets, and possible actions. Over the last 20 years the security research and standardization have proposed a variety of taxonomies for adversary classification based on systematization of psychological studies or security models [234, 235, 236, 233, 237, 238] or driven by the needs of a security analysis methodology [239, 66, 240]. As we found out, most taxonomies in our review operated with similar categories and description patterns. An adversary model in contrast to a profile uses a set of attributes and values to characterize an adversary and to define the constraints, e.g., limited resources or skills. The contents of an adversary model are not standardized as well and may vary depending on a system and goals of the analysis. The work [233] reviewed adversary models for cyber-physical systems and proposed a generalized taxonomy of adversary attributes.

The reference security standard DIN VDE V 0831-104 [66] does not detail adversary profiles. Instead it uses a set of qualitative dimensions (resources, knowledge, motivation) and their possible combinations as preliminary security levels. Thus, in order to compare adversary taxonomies and identify potential gaps, we first review adversary profiles found in the related work. To provide a more structured presentation of the adversary profiles we follow a template for a threat community introduced in the FAIR methodology[2] [241].

Threat analysis in FAIR considers any threat agent as part of one or more threat communities being defined using the following characteristics [241]:

- Motive, e.g. ideology
- Primary intent, e.g. damage
- Sponsorship, e.g. unofficial
- Preferred target:
  - Preferred general target characteristics, e.g. supporters of an opposite ideology
  - Preferred specific target characteristics, e.g. high visibility
  - Preferred targets, e.g. transport infrastructure
- Capability
- Personal risk tolerance
- Concern for collateral damage

In the following, we compile several adversary profiles based on our taxonomies overview [234, 237, 235, 236, 239, 233, 238, 240] and the threat community template [241] and discuss their relevance in the railway context. Though we aim for a characterization as comprehensive as possible, we do not claim it to be complete or perfect. Our primary goal is to develop a structured understanding of potential threat sources for railway CCSs.

[2] Factor Analysis for Information Risk (FAIR) is an open standard by The Open Group for the information risk management model and is listed by NIST as a complementary standard to other methodologies such as ISO/IEC 27002:2005, OCTAVE, etc. for quantifying and prioritizing risk (see `http://www.fairinstitute.org`)

**Novice (newbie, script kiddie, basic user, lamer, unstructured hacker)** category represents a non-sophisticated non-professional adversary with very limited skills in programming or attack methods and tools as well as very limited knowledge about target systems. A novice is driven mostly by intellectual factors like curiosity, thrill, or boredom, as well as desire for recognition. They have very limited resources and depends entirely on exploits and rootkits readily available on the Internet. A novice uses these tools as they are, mostly without understanding how they work or what consequences can be expected. The choice of a target is also mostly determined by available attack tools, i.e. it is more or less random. For this reasons, attacks of this kind are usually easy to detect with standard provisions. This puts a novice at a high personal risk (of being caught and punished). But the lack of knowledge and carelessness of this adversary can put potential target systems at a high risk of a serious damage (up to destruction), too. It just takes a good quality exploit disseminated by a more skilled or targeted adversary. Novices have plenty of time to search and try a wide range of tools and targets and can collaborate in small ad hoc groups. But they would lose interest fast, if an attack does not succeed at once.

The evidence that railway systems can attract this category of adversaries dates back to 2008, when a teenager hacked into a Polish tram system and managed to change points and derail several vehicles using a modified TV remote control[3]. In case of the CCS, a novice may target unprotected track side components (OCs or field elements) as well as connected CCS components and network services, especially those exposed to the Internet, if they demonstrate known vulnerabilities exploitable with readily available tools.

**Cyber punk (crasher, thug)** category represents a non-professional adversary with limited skills aiming for publicity, recognition and sometimes profit. Cyber punks are moderately malicious in the sense that they choose high profile targets guaranteeing strong media attention and an attack method producing visible effects that may involve a serious damage to a target system, though it is not a primary intent of this category of adversaries. In order to carry out an attack, a cyber punk can do small changes to or combine existing exploits and rootkits to tailor them to the target as well as develop own non-sophisticated tools. Adversaries of this category have limited resources in terms of financial support or expert knowledge, but can invest sufficient time into developing an attack on a particular target and can collaborate with each other in small ad hoc groups or communities.

Delays and disruptions in the rail transport system, especially those in passenger transportation services, often cause public outrage and are widely reported in the news, whether being caused by interlocking and signaling problems[4], [5], and server failures[6], or by hacked public information boards, video screens[7], and ticketing systems[8]. This close media attention is exactly what cyber punks seek to achieve. In order to disrupt rail service and cause delays a cyber punk may target open track side components of the CCS such as object controllers or field elements as well as connected CCS components and services, especially those exposed to the Internet, if they have known and easily exploitable vulnerabilities to e.g. push messages, manipulate signals, etc.

**Hacktivist (mercenary)** category represents political activists pursuing certain political agenda and targeting organizations for political reasons. One of the most prominent examples is the hacktivist network Anonymous. Collaboration for a common goal is characteristic for this category and provides for considerable human resource for targeted attacks. Financial support, on the contrary, is relatively low. Hacktivists may seek to break into IT systems to obtain confidential information that can harm their target and make it public on the Internet or organize massive denial-of-service attacks against the target's website or infrastructure. The primary intent in this case is to cause reputational damage to the organization that opposes hacktivists' ideology and potentially disrupt its activities. In order to do so, hacktivists might use free exploits from the Internet or develop own unsophisticated tools, to realize e.g. DoS or defacement attacks against websites and social media accounts or infect the target with malware.

Rail transport comes at times into focus of political activists and hacktivists too. For example, signaling equipment has been recently set on fire in the face of G20 meeting paralyzing railway tracks in several federal states across Germany[9]. Another example of disruptions caused by political activists is Greenpeace sabotaging the railway to prevent

[3] www.telegraph.co.uk/news/worldnews/1575293/Schoolboy-hacks-into-citys-tram-system.html

[4] www.bbc.com/news/uk-wales-39757840

[5] www.focus.de/regional/koeln/deutsche-bahn-verspaetete-zuege-wegen-stellwerksstoerung-in-koeln_id_6317196.html

[6] www.tagesspiegel.de/wirtschaft/am-ostermontag-schwere-stoerung-bei-der-bahn/19678476.html

[7] ccrail.com/washington-union-station-video-screens-hacked-and-rush-hour-passengers-shocked

[8] mashable.com/2016/11/27/san-francisco-muni-hacked/#7Mhz8O4TAEqA

[9] www.dw.com/de/anschl%C3%A4ge-auf-bahnanlagen-deutschlandweit/a-39303160

the transportation of nuclear waste[10]. Notably, in both cases only simple physical attacks took place.

**Criminal (petty thief, organized crime, criminal group)** category represents professional and relatively skilled adversaries who engage in cyber attacks for financial gain. Criminals target IT systems that process and store information, such as bank accounts, credit card details, personal data and intellectual property that can be easily monetized (e.g. sold) or used in further criminal activities like online fraud, identity theft, extortion, phishing, spamming, copyright infringement, or money laundering. Criminals can develop advanced knowledge of network attacks to e.g. hack into e-commerce applications or a database and can invest significant financial resources, e.g., to hire professional hackers or buy special-purpose attack tools, if they expect high return on "investment". Criminals often work in organized groups and coordinate their efforts. This category of adversaries seeks to keep criminal activities hidden, partially, because publicity can affect the profit (e.g. credit cards can be timely blocked) as well as lead to legal prosecution, so a major system failure or breakdown is not their primary intent and in general undesirable.

Considering that the CCS does not handle financial or personal information, it is unlikely to be targeted by criminals. On the other hand, spyware and malware distributed by criminals can pose a potential threat to servers and workstations in the rail operations center, in case they use standard or not sufficiently protected software.

**Cyberterrorist** category represents an intentional malicious adversary driven by political or ideological motivation. Cyberterrorists seek to destroy, massively disrupt or damage IT systems and infrastructures and thereby cause major harm or mass casualties in order to induce fear and trigger off panic among certain groups of people or put pressure on governments [242, 243, 244]. This category is the only one deliberately aiming for collateral damage with their attacks and least concerned about personal risk or stealthiness of an attack. Moreover, wide publicity and extensive media coverage associated with this kind of incidents is what cyberterrorists look for. Cyberterrorists usually demonstrate limited skills with regards to programming, attack techniques or system knowledge and rely on tools that they can find or buy in the Internet, such as spyware or malware to disrupt system availability. More seldom, cyberterrorists can advance to the level, where they can modify or create basic tools [244]. On the other hand, this category of adversaries has considerable funds at their disposal, e.g., raised from illegal activities, to buy exploits and hire or recruit members with required expert knowledge in order to plan and organize an attack. Similar to ideologically motivated hacktivists, cyberterrorists mostly operate in groups with varying level of organization.

Reports on railway security and safety indicate the growing number of incidents due to terrorist acts [245, 246]. Though the reported cases listed only physical attacks, ongoing digitalization of the rail infrastructure might change it. IT attacks can be more favorable for terrorists, because compared to physical attacks they can be carried out with lower personal risk (remotely, anonymously, without personal sacrifice), require less resources (no weapons, etc.), and have a potential to affect more people [242, 244]. The recent Petya/NotPetya attack[11] on the energy infrastructure in Ukraine supposedly aiming to destroy as much data as possible may give an idea of a potential impact scale.

**Malware author (coder, virus writer)** category represents very skilled but non-professional adversaries, who do not carry out offensives against IT systems or infrastructures, but instead develop and publish malware such as exploits, rootkits, worms, trojans as an intellectual exercise. Malware authors cannot be considered malicious because their primary motivation is to gain influence and recognition, not to compromise any specific target. Less skilled adversary categories like novices, cyber punks, hacktivists and cyberterrorists benefit from the tools created by a malware author actively using them to attack real IT systems and infrastructures.

This category is relevant for the CCS because of potential uncontrollable spreading of such not target-specific viruses and worms. For example, in 2003 a "Sobig" virus spreading via e-mail hit the computer system at CSX Corp.'s Jacksonville, Florida headquarters, shutting down signaling, dispatching and other systems and caused an outage in the entire CSX system in 23 states[12]. Deutsche Bahn fell victim to the WannaCry attack, ransomware spreading via a vulnerability in Microsoft's implementation of the Server Message Block (SMB) protocol[13].

**Ethical hacker (white hat hacker, old guard)** category represents highly skilled non-malicious adversaries, who actively attack IT systems and infrastructures in order to discover vulnerabilities in widely used technologies, protocols and

[10] www.theatlantic.com/photo/2011/11/protesters-disrupt-german-nuclear-waste-shipment/100196/

[11] www.heise.de/security/meldung/Petya-NotPetya-Kein-Erpressungstrojaner-sondern-ein-Wiper-3759293.html

[12] www.cbsnews.com/news/virus-disrupts-train-signals/

[13] www.telegraph.co.uk/news/2017/05/13/cyber-attack-hits-german-train-stations-hackers-target-deutsche/

applications and eventually help the community to fix them [238]. Ethical hackers are primarily motivated by curiosity, intellectual challenge or respect/reputation and usually publish their findings (e.g., by responsible disclosure) and make tools available for the interested parties [235, 234]. Due to a deep technical understanding of IT and versatile methods of attack on networks, systems or applications, ethical hackers create own customized attack tools (exploits, scripts, rootkits, etc.) and prefer manual strategies to automated ones [238]. Being IT professionals, ethical hackers have average financial resources and can afford themselves state-of-the-art (SOTA) software and hardware to tamper with, but rarely specialized industrial-level systems. They may be professionally engaged in penetration testing or vulnerability assessment of IT systems and infrastructures or do research in the area. Active communities and continuous information exchange is also quite common for this category.

Security of critical infrastructures such as railway are highly interesting to the (ethical) hacker community, which is proven by programs of the Chaos Communication Congress and DEFCON. Ethical hackers are unlikely to seriously damage or destroy operating IT systems and infrastructures, provided that the security issues they discover are properly and timely fixed by the owners. Like in the case of the malware authors, less skilled adversaries who might be driven by different motivations benefit from the methods and tools created by an ethical hacker.

**Professional hacker (black hat hacker, cracker)** category represents highly skilled malicious adversaries motivated by personal or financial gain [235]. They exploit software or hardware vulnerabilities to break into IT systems and infrastructures for profit or revenge, e.g., to steal and sell valuable information, discover and sell vulnerabilities and exploits (to target organizations or on black market), to ransom, or to simply disrupt authorized usage or otherwise damage their target. Botnet operators are also professional hackers who compromise and take control over multiple networked (usually internet connected) systems and use them for distributed denial-of-service attacks, spam and malware distribution, as well as bitcoin mining [240] or offer them as a so-called rent-a-botnet service to others. Black hats have access to SOTA hardware and software, including available domain-specific solutions if they are not overly expensive, and can rely on sponsorship from foreign governments (e.g. for industrial espionage) or organized crime [234]. Similar to ethical hackers, professional hackers prefer self-made tools and manual strategies for attacks, in contrast to them, black hats act with a high degree of covertness and usually work alone [238].

Still, some incidents make it into into the media. Reportedly, in November 2016 the ticketing service of San Francisco's public railway system was interrupted for two days due to malicious software in a computer network. Later, the hackers threatened to release 30 Gigabytes of supposedly stolen employee and customer data unless certain ransom is paid and the vulnerability is fixed[14]. Thus, one can consider several incentives for a black hat hacker to tamper with a CCS network: (i) to paralyze rail traffic and blackmail a respective railway operator or a government; (ii) sell vulnerabilities and exploits on black market; (iii) create a botnet.

**Nation-state (foreign intelligence service, government agent hacker, military hacker, information warfare)** represents the most powerful adversary category with very high skills, resources and motivation [233]. Nation-state (or state-sponsored) adversaries are driven mostly by patriotic motives and focused on economic and industrial espionage and so-called "information warfare", i.e. carrying out offensives against a foreign nation in order to disrupt its communication channels and networks or hijack them for disinformation, destabilize economic infrastructures (markets, banks, etc.) and disable supply networks (energy, transportation, etc.) [238, 240, 234]. An unambiguous attribution of an attack in this case could trigger a well justified response from the victim state and lead to an undesirable conflict escalation[15]. For this reason, a nation-state adversary always acts stealthily seeking to minimize the chance of attack detection and hide or falsify any possible traces to make it impossible to identify the source. Such kind of highly sophisticated and long-term targeted attacks is often referred to as an advanced persistent threat (APT) [247]. In contrast to other categories, where the least protected system is attacked, an APT adversary sticks to a specific target and will try out all sorts of strategies to bypass its protections and get control over it. A groundbreaking example of an APT is Stuxnet, "the world's first digital weapon[16]", which targeted an uranium enrichment plant in Iran to sabotage the centrifuges in 2008-2010 [248].

The railway infrastructure including CCS is very likely to face APT [249]. The security of railway communication networks might be undermined by some backdoor surveillance functionality, such as one suspected to be built-in in Chinese telecommunication devices[17] or one reported in 2010 due to an unauthorized field upgrade to Cisco hardware

[14] fortune.com/2016/11/28/muni-hack-san-francisco/

[15] resources.infosecinstitute.com/cyber-warfare-from-attribution-to-deterrence/

[16] www.wired.com/2014/11/countdown-to-zero-day-stuxnet/

[17] www.theguardian.com/technology/2012/oct/08/china-huawei-zte-security-threat

routinely made by NSA[18]. Recently, another large-scale targeted attack on computer networks of energy facilities and manufacturing plants operators was reported and attributed to an APT actor[19], that could have an impact on the railway infrastructure too.

**Malicious insider (trusted insider, privileged insider, internal, disgruntled employee, user malcontent)** category represents a skilled malicious but non-professional adversary targeting IT systems and infrastructures of the employer organization for revenge, i.e. to harm the employer, or personal gain [234, 237, 235]. Insiders are trusted employees who misuse their job-related access to organization's internal resources or secrets. They are in a position to bypass defenses and steal confidential data, install malware or spyware, disclose confidential information like system configurations or user credentials, sabotage systems and equipment, and create safety hazards [250]. It is the only category of adversaries that has an inside knowledge about a targeted infrastructure and can achieve their goals without need for any special adversary tools or techniques [233, 240]. The capability of insiders depends on their role in the organization. Cleaning staff and technicians may have physical access to restricted areas and equipment, normal users may access sensitive information or functions through dedicated applications and services, privileged users such as system administrators manage whole IT infrastructure including security systems and have almost unconstrained access rights [251]. Privileged insiders also have higher technical skills and can cast more complex attack strategies to minimize the risk of being caught. Insiders usually act alone and do not have any financial support, unless engaged with a nation-state adversary, industrial spies or organized crime.

The American Public Transportation Association (APTA) confirms that it is a challenge to protect railway systems against malicious insiders [252]. There are numerous examples of insider threat incidents, many of them involving critical infrastructures [253, 254].

**Supplier (service provider, vendor, support)** represents highly skilled non-malicious adversaries who deliberately manipulates the hard- or software (e.g. by embedding backdoors or installing spyware) that they provide or maintain and this way make it possible to compromise security of IT systems or infrastructures built on it [251]. Sometimes suppliers are regarded as insiders, because they may have physical access to an organization's internal systems. A supplier has a deep understanding of own products and might also know details of the IT infrastructure where they are supposed to be deployed. Undesirable (and undocumented) functions can be introduced during production by qualified staff and financed by a supplier's company itself (or by a nation-state). The motivation of the adversary might be complex: (i) compliance with the State guidelines, such as requirements to enable state-run trojans for law enforcement in COTS enabling communications and networking or to provide tools for information warfare; (ii) monitor usage and collect data in the background that can be used to exclude or limit seller liability, in a conflict of interests, or for other business purposes; (iii) manipulate certain parameters or test values to their benefit, for example, to show compliance to regulations as it happened in the recent "dieselgate" affair. Any backdoor functionality – even if introduced for legitimate purposes – can be misused by other categories of adversaries to compromise the system. Another way for a supplier to create a potentially dangerous situation is to share data such as monitoring logs with third parties without proper anonymization.

For example, the American Public Transportation Association (APTA) recognizes this adversary category as one of the possible ways how malware can be introduced into an IT system and also suggests that (potentially insecure or unpatched) tools or applications used by a supplier, e.g., to configure or maintain its equipment should be removed or disabled unless necessary [252]. For these reasons, it is also recommended to define a dedicated vendor access policy to isolate vendors and suppliers from an organization's confidential data and critical systems outside their responsibility [255].

## 7.3. Risk Analysis using DIN VDE V 0831-104

In order to derive security requirements, we used the approach described in the German prestandard DIN VDE V 0831-104 for security in railway signaling networks [66]. This guideline adapts the IEC 62443 framework for industrial control system security to railway signaling and relevant railway safety standards, such as EN 50128 [256], EN 50129 [257], and EN 50159 [258].

[18] www.theguardian.com/books/2014/may/12/glenn-greenwald-nsa-tampers-us-internet-routers-snowden

[19] www.nytimes.com/2017/07/06/technology/nuclear-plant-hack-report.html

### 7.3.1. Adversary Model

In order to describe capabilities of an adversary, DIN VDE V 0831-104 uses only three dimensions: resource, knowledge, and motivation (cf. [233]). The *resource* dimension reflects the financial and workforce capacity of the adversary to prepare and launch an attack. The *knowledge* dimension describes what the adversary knows about the system before attacking it and can use to create opportunities for a successful offensive. These dimensions are characterized by numerical values from low (2) to high (4), where adversaries with basic capabilities (1) are not considered [66]. Thus, *generic* knowledge (K $= 2$) comprises of publicly available information such as protocol specifications and COTS hardware. An adversary with *extended* knowledge (K $= 4$) has access to some closed information usually only available to a small circle of experts (insiders) working with the system. *Low* financial resources (R $= 2$) comprise a few thousand Euro while *extended* financial capacity (R $= 4$) provides resources in the magnitude of state actors.

In order to describe the adversary's motivation DIN VDE V 0831-104 introduces railway-specific mitigation factors related to the risk for the adversary to be discovered. The higher this risk for a particular attack, the lower is the motivation to carry it out. This way, existing security controls can be taken into account. The standard considers the following mitigation factors: location (LOC), traceability (TRA) and extent of attack (EXT) [66]. LOC determines whether an attack can be executed remotely (LOC $= 0$) or only given a physical access to the system (LOC $= 1$). Remote attacks are considered to be more dangerous, as the chance for the adversary to remain undiscovered is higher. Similarly, if an attack can be traced to the adversary (TRA $= 1$), it is less dangerous than an untraceable attack (TRA $= 0$). EXT denotes the potential damage of an attack, which can be either *critical* (EXT $= 1$) or *serious* in case fatalities might be involved (EXT $= 0$).

This adversary model is applied during the risk assessment to evaluate each potential threat and determine the security level required to protect the system against it. Though this approach does not explicitly employ traditional adversary categories including, e.g., basic user, cyber-criminals, terrorists, insiders, or nation state [233], it uses the same idea to describe adversary's capabilities.

In the specific context of this work involving safety aspects, the only relevant goal of the adversary is to cause collisions or derailment of trains. This goal can also be achieved by changing the color of a light signal, or detaching the light signal from the OC and applying current, so that the signal shows clear when it should show stop. For example, political activists from Greenpeace are reported to cause disruptions by sabotaging the railway to prevent the transportation of nuclear waste. This kind of simple physical attacks targeted against individual field elements have always been possible and cannot be fully prevented by IT security mechanisms.

The digitalization in turn introduces much broader adversarial opportunities due to connectivity and usage of regular IT components. Thus, our goal is to protect the railway signaling network against large-scale attacks, which can be applied to a multitude of field elements at the same time and can paralyze the complete infrastructure. In contrast to such scenarios, physical attacks are much more restricted. As the hardware in `Z.ILS` and `Z.MDM` is installed in a building with physical access control, it is reasonable to assume that an adversary cannot gain physical access to this hardware to modify or replace the components. The OCs in `Z.OC` are installed in a junction box and therefore physically accessible for the adversary. However, we assume that physically tampering with an OC is equivalent to local attacks, we described before (blocking the tracks, changing colors of light signal). They do not scale, because each field element has to be attacked individually. Also it is considered virtually impossible to protect against such physical attacks in a large-scale infrastructure like nationwide railway signaling [259, 260]. For this reason, we focus on the protection against high-impact large-scale attacks.

### 7.3.2. Threat Analysis and Definition of Security Levels

According to DIN VDE V 0831-104, the next step before to define security requirements is to perform threat-based risk analysis for each component and zone in our system model.

During this analysis, threats are related to the Foundational Requirements (FRs) of IEC 62443, i.e., identification and authentication control (IAC), use control (UC), system integrity (SI), data confidentiality (DC), restricted data flow (RDF), timely response to events (TRE), and resource availability (RA).

In order to estimate the risk associated with a threat, the adversary capabilities in accordance with the adversary model from Section 7.3.1 are used. The adversary's resource and knowledge capabilities are combined to form a Preliminary Security Level (PSL) related to the given threat. Thus, each threat is assigned values for the adversary capabilities (R, K) and values for the mitigation factors (LOC, TRA, EXT) to calculate the PSL. The PSL is later used to calculate the final Security Level (SL) for the respective threat. A SL ranging from 1 (low) to 4 (high) describes the level of protection a system provides against the adversary.

Examples of the identified threats with assigned values for the adversary are provided in Table 7.1. For example, `T.SI.Attacker.Malware` describes an adversary, who introduces malware to undermine the integrity of the railway CCS. `T.RA.Attacker.DoS` covers DoS attacks.

Table 7.1: Threat Examples for Railway Signaling

| Threat | FR | R | K | PSL | LOC | TRA | EXT | SL |
|---|---|---|---|---|---|---|---|---|
| `T.SI.Attacker.Malware` | SI | 3 | 4 | 4 | 0 | 0 | 0 | 4 |
| `T.RA.Attacker.DoS` | RA | 2 | 2 | 2 | 0 | 0 | 1 | 1 |

As defined by DIN VDE V 0831-104 [66], the final SL, using the PSL and the mitigation factors, is calculated using the following equation:

$$\text{SL} = \text{PSL} - \max\{\text{LOC, TRA, EXT}\}. \tag{7.1}$$

According to the equation, the value of the PSL is reduced by one, if any of the mitigation factors equals one (corresponding to a logical “or”). This means that a threat that can only be executed locally (LOC), is traceable (TRA), or has only a critical extent (EXT) will reduce the PSL by one level to form the SL.

Table 7.2: SL Vectors of the Zones

| Zone | IAC | UC | SI | DC | RDF | TRE | RA |
|---|---|---|---|---|---|---|---|
| `Z.OC` | 3 | 2 | 4 | 1 | 1 | 3 | 1 |
| `Z.ILS` | 3 | 2 | 4 | 1 | 1 | 3 | 1 |
| `Z.MDM` | 3 | 2 | 4 | 1 | 1 | 3 | 1 |

This calculation is done for each of the identified threats in the seven FRs. To calculate the SL value for the three zones, each is assigned a vector of seven values corresponding to the respective FRs, as shown in Table 7.2. The seven entries are determined by the maximum SL value over the threats assigned to the respective zone and FR. For each zone, the FR with the greatest SL determines the security level of the zone. For simplicity, we write SL $x$, when we refer to an SL vector with maximum entry $x$. In this way, the SL vector of `Z.OC` $(3, 2, 4, 1, 1, 3, 1)$ yields SL4 for the zone. Respectively do the vectors of `Z.ILS` and `Z.MDM` yield an SL of 4.

We identify three decisive threats that determine the security levels. We use them to exemplify how the adversary capabilities and mitigation factors lead to the SL. Two of the decisive threats are responsible for the value of 4 in all three zones. The first decisive threat is the remote execution of malware on the systems in our reference architecture (`T.SI.Attacker.Malware`, see Table 7.1). We assigned an adversary with moderate resources (R $= 3$) and extended knowledge (K $= 4$) to it, resulting in PSL $= 4$ for the threat. The assessment of the mitigation factors resulted in the following values: the threat description implies that it is remotely executable (LOC $= 0$). A skilled adversary is assumed to be able to hide the traces, such that we consider the threat as not traceable (TRA $= 0$). Also, an adversary with deep knowledge (K $= 4$) is capable of performing a carefully targeted attack with potentially serious extent (EXT $= 0$). Thus, none of the mitigation factors apply to reduce the final SL. Subsequently, adding the mitigation factors and the PSL into Equation 7.1 yields SL4.

The second decisive threat describes the manipulation of patches such that legitimate processes execute malicious code chosen by the adversary when rolled out to devices through update mechanisms. Analogous considerations for adversary capabilities and mitigation factors as in the previously discussed case apply to this threat leading to SL 4.

Only for zone `Z.MDM` there is a third decisive threat that results in SL 4. It covers the manipulation of data on the MDM where the firmware of the ILS and OCs are stored and distributed from. This threat poses a high risk, because the firmware can be manipulated remotely at a central point from which it is distributed to unsuspecting network entities. Without further checks, the ILS and OCs of an entire station's signaling network can be compromised. Again, we consider an adversary with R3 and K4 to perform this attack and could not identify an applicable mitigation factor (LOC $= 0$, TRA $= 0$, EXT $= 0$). Hence, the analysis of this threat also leads to SL 4 for zone `Z.MDM`.

## 7.4. Security Requirements Elicitation

After having derived the SL of each zone, the specific system requirements can be retrieved from IEC 62443-3-3. The standard contains a list of 100 system requirements that are applicable to each zone depending on the identified SL. We evaluated the SL for each FR to select the security requirements from the IEC 62443-3-3 standard. As a result of our risk analysis, we found 69 system requirements that are relevant for our system model.

Since it is a standard for Industrial Automation and Control System (IACS), the requirements of IEC 62443 are very generic. In order to reduce the complexity, we choose to explore an additional approach to derive security requirements. For this approach, we elicit requirements using the methodology [261], which proposes to transform each threat identified during risk analysis to a security requirement. This also opens the possibility to define additional railway-specific requirements, which are not reflected IEC 62443. The results of our requirement elicitation process are presented in Table 7.3, where $R_1$, $R_{11}$, and $R_{13}$ are examples for railway-specific requirements.

Table 7.3: Security Requirements for Railway CCS Architecture

| | |
|---|---|
| $R_1$ | The system shall detect unauthorized physical access to its subsystems and/or prevent relevant exploitation of physical access |
| $R_2$ | The system shall not allow the compromise of a communication key |
| $R_3$ | The system shall not disclose classified or confidential data to an illegitimate user |
| $R_4$ | The system shall exclude compromised endpoints from communication |
| $R_5$ | The system shall not use insecure transfer methods |
| $R_6$ | The system shall not allow any unauthorized user to access an endpoint |
| $R_7$ | The system shall not allow unauthorized and unauthenticated communication between endpoints |
| $R_8$ | The system shall not violate the run-time behavior requirements |
| $R_9$ | The system shall allow for the updating of security mechanisms, credentials, and configurations to patch known vulnerabilities |
| $R_{10}$ | The system shall not allow the execution of unauthorized software instances |
| $R_{11}$ | The system shall maintain the transmission system requirements defined in EN 50159 |
| $R_{12}$ | The system shall provide means to detect an undesirable system state change and anomalies |
| $R_{13}$ | The system shall impede that an unauthorized user can force it into one of the fall-back levels defined by the railway safety process |
| $R_{14}$ | The system shall maintain the integrity of software, firmware, configuration, and hardware |

In the following, we discuss those requirements that are specific to railways as they could violate the safety constraints posed by the domain standards.

The physical access detection required by $R_1$ is especially relevant for the railway domain, because OCs are spatially distributed over large areas next to railway tracks and cannot be as well protected as, for example, within factory premises. Therefore, unauthorized physical access to the junction box of the OC has at least to be detected such that further actions can be triggered, e.g., by a security operations center to avoid or mitigate consequences.

In order to keep a railway station operational, $R_4$ requires that compromised endpoints (OCs) can be excluded from the network, such that benign OCs are not affected. An endpoint is considered compromised, if an adversary can remotely control the safety functionality, because the OC accepts the adversary's commands or the adversary controls the safety-critical software on the endpoint. The disclosure of cryptographic keys belonging to an endpoint renders it compromised as well. Physical access to the OC's hardware constitutes a compromise that can be detected if the junction box is opened or the OC is removed from the rack. Physical access to the steered field element is not a compromise as this attack is already possible in current railway infrastructures without digital components. However, physical attacks of this kind do not scale to multiple OCs, if no shared secrets can be gained by the adversary.

Requirement $R_5$ excludes transfer methods such as network protocols that involve cryptographic functions which usage is discourage by institutions like NIST or national agencies for information security. The requirement enforces the usage of communication protocols that do not employ cryptography that is considered broken.

Due to safety reasons, railway signaling networks require a failure disclosure time, which is addressed by $R_8$. Any security measure that influences the network traffic (e.g., message encryption) must not exhaust the network resources such that the network latency exceeds a threshold of 50 ms as specified by railway safety standards and railway operator specifications.

A common requirement from both security and safety perspectives is the robustness of a transmission system against repetition, deletion, insertion, re-sequencing, corruption, delay, and masquerade of messages. This is specified by EN 50159, a European standard for safety-related communication in transmission systems required to receive admission to operate a railway system. $R_{11}$ ensures that these requirements are fulfilled and also considered in the design of a security architecture to avoid fulfilling a requirement twice. Some security functionalities might already be available in the system due to fulfilling EN 50159 or can be established by adding only minimal features.

During the design of the security architecture, it must be considered that railway signaling has processes that take effect in case of technical failure in order to maintain operation of the railway system. These fall-back processes involve human interaction and do not provide the full extent of safety and capacity compared to a fully functional, automated interlocking in terms of failure rate. This risk is covered by $R_{13}$, which requires the security architecture to be designed in such a way that it does not allow an adversary to force the interlocking system into a fall-back state. The attack surface should not be increased by the security architecture compared to attacks with the same effect already possible today, e.g., physically destroying a cable. A security architecture that can force the safety system into a fall-back state enables the adversary to remotely implement a large scale DoS attack causing major disruption in the railway transportation system.

## 7.5. HRA Security Architecture

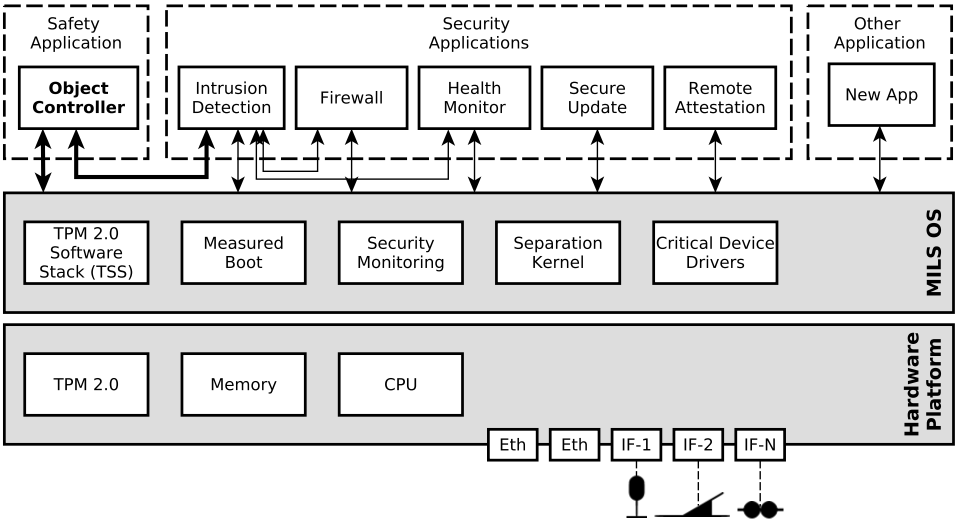


Figure 7.2: Haselnuss Security Architecture for Secure Object Controller

Figure 7.2 shows our proposed architecture, Haselnuss Reference Architecture (HRA). It consists of three main components: a hardware platform with a hardware security module in the form of a TPM 2.0, a MILS SK, and various security applications. The TPM serves as a security anchor and enables, among other things, secure storage of cryptographic keys, e.g., to secure communication connections, measured boot to record software executions in a tamper-evident manner, and remote attestation to allow authorized external parties to detect tampering with the system software. The certifiable MILS SK allows the joint operation of safety and security applications on the same hardware. The SK controls critical hardware interfaces and ensures the non-interference and the resource availability for a safety application. In our case, the safety application is a digital OC for NeuPro. Security applications are, e.g., anomaly detection methods which detect attacks over the network, secure software update protocols, or a classic firewall. Possible applications are not limited to these examples, the integration of further safety- or security-relevant sensors located on the tracks can also be enabled this way as shown in the study [24].

**Hardware Platform.** The OC is located remotely at railway tracks. To ensure that this safety-critical system is not manipulated by malicious adversaries, a continuous proofing is performed using Trusted Computing technologies [262].

As a hardware Root of Trust (ROT), a TPM 2.0 specified by the Trusted Computing Group (TCG) [114] is used. It provides secure storage, secure execution for cryptographic operations, as well as additional features, e.g., facilitating authenticated boot and secure update procedures. The TPM is a special type of a hardware security module. In our case, it is implemented as a dedicated chip. It was shown that cryptographic agility, enhanced authorization and modularity of the TPM 2.0 specification makes it suitable to secure long-lived resource-constrained systems like controllers [120]. Authenticated boot and remote attestation build on the capability of the TPM to persistently store the status of the loaded software as so-called configuration measurements (hash-chains) within the TPM's Platform Configuration Registers (PCRs) [263].

To further protect the OC against physical attacks, the housing or chassis of the hardware platform raises an alert every time it is opened. The alerts are written into the TPM via the BIOS and can only be reset by authorized personnel. Chassis open alerts work even if the OC is disconnected from power supply and make physical interference from an adversary evident.

For the redundant network connection, the hardware platform provides two Ethernet interfaces. In order to steer the field elements, wired interfaces to signals, point machines and train detection systems are designated.

**Software (MILS) Platform.** The software platform is constructed following the MILS methodology [264]. The security core of the proposed architecture builds on the SK. The MILS platform architectural layer provides a controlled interface to the hardware and enforces security policies for information flow, access control, and resource availability.

The TPM Software Stack (TSS) 2.0 specified by the TCG [114, 265, 266] provides an API for TPM 2.0 to applications and is also part of the platform. A software for critical devices controls direct access of hardware to the main memory without involvement of Central Processing Unit (CPU) and Memory Management Unit (MMU) that would otherwise bypass the SK. A Security Monitoring Infrastructure module inside the MILS platform collects information about system state and network traffic required for detecting the network anomalies and performing health monitoring.

**Security Applications.** Security functions necessary for the OC to fulfill the identified security requirements are realized by security applications running on the MILS platform. This includes applications enabling the protection of the OC software integrity at boot- and runtime, integrity reporting, remote attestation, and secure software update as well as health monitoring, network traffic filtering, and intrusion detection.

**Safety Applications.** Alongside the security applications, one or more safety applications run on the MILS platform. In our case, we encapsulate the existing functionality of an OC in the safety application that operates a point or a signal.

### 7.5.1. Hardware Trust Anchor and Security Protocols

A TPM 2.0 provides a hardware ROT. Using an authenticated boot procedure, the platform's ROT for Reporting generates a report about the boot process by creating evidence about the integrity of the booted software components. Using a remote attestation procedure, this evidence can be conveyed to an external verifier in order to appraise the evidence.

For example, the Time-Based Uni-Directional Attestation (TUDA) protocol [267] can be used as remote attestation procedure. In contrast to traditional remote attestation procedures (e.g., [268]), the attestor (or the verifier, respectively, depending on which entity initiates the procedure) does not need to send a nonce, in order to prove the freshness of the created evidence, but uses trusted time-stamps originating from an external trusted source of time. This approach has the advantage of decoupling the activities of creating the integrity evidence, conveying it and then appraising it via a verifier; therefore enabling the appraisal of past states and also reducing the utilization of the TPM significantly.

### 7.5.2. Security Architecture for Authenticated Boot

In order to create evidence that enables integrity proving of a OC and its corresponding software components that compose the runtime environments, an authenticated boot procedure is introduced. Even before booting, integrity measurements – resilient hash-values of every software component involved, including the BIOS – are created and aggregated in a corresponding partition of the OC. Measurements of safety applications and their configuration data are created with the help of the introduced authenticated boot kernel component (see Figure 7.2) on startup and during observation of update processes. The separation enforced by the SK guarantees that the creation of integrity evidence, as a basis for remote attestation procedures, has no impact on the runtime behavior of the safety applications.

The relationships of interfaces among the entities in the OC is illustrated in Figure 7.2. Basically, every higher level interface with respect to creation of evidence is based on the TSS 2.0. Secure communication channels between partitions are realized using communication objects that are governed by the SK. The SK provides the separation assurance required to shield the safety-related partitions from malicious or unintentional harmful use of these communication objects.

### 7.5.3. Runtime Integrity Monitoring

Considering that re-booting or updating an OC does not happen frequently, runtime mechanisms for integrity control and system resilience are required to fulfil the security requirements. This functionality is provided by a dedicated health monitor (see Figure 7.2) that continuously performs collection of data regarding application state, analyses this data for changes or anomalies and does reporting [269, 270]. For this purpose, the communication objects of the SK need to be extended with monitoring capabilities to detect failures or attacks in system safety or security services. Another data source for the health monitor is the kernel-level security monitor. The health monitor can be non-invasive or enable invoking specified resilience mechanisms similar to the ones proposed in [271, 272] desired for some particular scenarios. Thus, to reduce the service downtimes, a OC may be able to recover after failures or attacks, e.g., by automatically resuming to a known valid configuration instead of failing. In this case, this behavior does not have any impact on the assurance level of the safety applications because the application itself is not changed by the health monitor.

### 7.5.4. Secure Update

Software, firmware, and configuration updates are necessary to fix software bugs and vulnerabilities in the OC. The new features introduced by the TPM 2.0 specification, such as monotonic counters and enhanced authorization, make it possible to protect system data, e.g., safety-relevant settings, and to ensure that only updates from authorized sources are accepted and downgrade attacks are prevented.

### 7.5.5. Intrusion Detection

The Intrusion Detection System (IDS) provides a defense mechanism against network-based attacks and applies after the network traffic has been initially filtered by a firewall. Types of network traffic are distinct and invariable. Protocols and ports for control commands, secure update, remote attestation are known in advance. The firewall can be configured by a predetermined whitelist. By collaborating among multiple OC instances in a defined area of the controlled field elements, the IDS is enabled to detect adverse commands, dangerous infrastructure configurations and misuse on application level. Our IDS solution allows the OCs to validate received commands as an additional layer of defense beyond authenticated communication channels. An ability current OCs do not possess. The IDS is fine-tuned to the Critical Infrastructure (CI)'s network topology (i.e., the track layout and the position of signals and points) and the utilized transport and application protocols. In this way, the IDS is enabled to leverage context information of the controlled infrastructure to enhance the intrusion detection accuracy. In a second step, counteractions on detected intrusions are defined that respect the safety functions of the CI. We carefully design the intrusive counteractions such that they do not alter the network channel properties beyond the specification of latency and message loss that the safety application is anyway required to tolerate.

## 7.6. Evaluation

### 7.6.1. Coverage of Security Requirements

In the following, we discuss how security requirements defined in Section 7.4 are covered by the proposed security architecture.

$R_1$ *The system shall detect unauthorized physical access to its subsystems and/or prevent relevant exploitation of physical access.*

Attack scenarios of physical access are addressed in several ways. General access to the chassis of the OC can be detected through the chassis open alert provided by the hardware platform. The access is reported using the authenticated TPM attestation means. Also the alteration of the main firmware of the device or the replacement of those hardware components that carry an Option-ROM can be detected by the attestation. Using the audit capabilities of the attestation protocol, even physical access to the device or management backend cannot alter post-incident logs in order to, e.g., fake the firmware or configuration status of the device during an accident.

Additionally, the devices prevent the cloning of a devices identity, since the relevant data is stored inside the TPM. Though a TPM and the associated identity can be stolen, they cannot be duplicated. The use of monotonic counters inside the TPM also prevent so-called roll-back attacks of local firmware, where older versions are used in place of more recent versions.

Please note that some physical manipulations are still possible – such as the tapping in between CPU and TPM. Even though concepts against such attacks exist (e.g. RayzorClam) they are not needed, since the same adversary class can reach results of direct I/O manipulation or hardware replacement much cheaper, than in-field PCB modifications.

$R_2$ *The system shall not allow the compromising of a communication key.*

The TPM 2.0 provides a secure storage for communication keys and any other cryptographic keys as well as a secure execution environment for cryptographic operations to protect them from compromise.

$R_3$ *The system shall not disclose classified or confidential data (such as access credentials) to any illegitimate user.*

Service access, user login and privilege escalation events can be stored in the TPM in order to create evidence about system state that may enable unauthorized disclosure and can result in actions ranging from event notification to automated remediation or quarantine procedures. The TPM 2.0 provides a secure storage for cryptographic keys, a secure execution environment for cryptographic operations and supports state-of-the-art cryptographic algorithms to protect confidential data from unauthorized disclosure.

$R_4$ *The system shall exclude compromised endpoints from communication.*

Continuous proving of platform integrity and attestation allows to detect and exclude compromised endpoints from the communication. Using a suitable integrity measurement architecture policy, those attestations can validate the integrity of code as well as data elements of the platform. The proposed architecture will measure all complete static partitions, including code and data segments. Additionally, an interface exists, that these measured (and well-behaved) partitions will in turn use to measure their dynamic (runtime) configuration data.

The concept of measuring code as well as data and doing so context dependently is not unusual; the default Linux policy will measure all code (executables and libraries loaded by all users). For the root user, it will additionally measure all data files opened. For example, measured runtime data can be the state of the controlled field element (e.g., signal aspect and direction of the point). The states are known in advance so that reference measurements can be created.

$R_5$ *The system shall not use insecure transfer methods.*

Our architecture does not use transfer protocols with known vulnerabilities. Update mechanisms are applied to react to newly discovered vulnerabilities (see R9).

$R_6$ *The system shall not allow any unauthorized user to access an endpoint.*

Access to the endpoints is protected by user credentials.

$R_7$ *The system shall not allow unauthorized and unauthenticated communication between endpoints.*

The TPM 2.0 can be used to equip each OC with a secure and unique identity that provides the basis for secure networking. Allowed network flows are assessed and enforced by a firewall.

$R_8$ *The system shall not violate the run-time behavior requirements.*

Separation mechanisms supported by the SK allow to statically assign user applications all resources needed for fulfilling its run-time requirements and provide guarantees that the safety application and security applications do not have to compete with each other for resources.

$R_9$ *The system shall allow for the updating of security mechanisms, credentials, and configurations in order to patch known vulnerabilities.*

The TPM 2.0 based secure update mechanism enables remote updates for security mechanisms, credentials, and configurations and rollback protection.

$R_{10}$ *The system shall not allow the execution of unauthorized software instances.*

Depending on the criticality of the software instance affected, an authenticated boot procedure enables countermeasures ranging from enabling restrictions on execution privileges to pausing affected partitions. A fine-granular action catalog ensures that countermeasures cannot affect functions of a safety-critical partition.

$R_{11}$ *The system shall maintain the transmission system requirements defined in EN 50159.*

An implementation of the architecture will be evaluated to assess that the requirements are not violated.

$R_{12}$ *The system shall provide mechanisms to detect an undesirable system state change and anomalies.*

Based on the collected integrity evidence, changes in the boot system configuration can be revealed. The health monitor and the intrusion detection analyze the runtime behavior of the applications and detect undesired states.

$R_{13}$ *The system shall impede that an unauthorized user can force it into one of the fall-back levels defined by the railway safety process.*

The applications of the security part of the shell concept (see [64]) operate free of interference with the applications in the safety part of the shell. Without substantial and well-defined reasons, the security does not trigger fail-safe states. The security architecture provides resilience mechanisms that help recover the system from attacks and reduce downtimes.

$R_{14}$ *The system shall maintain the integrity of software, firmware, configuration, and hardware.*

Boot and runtime integrity control of software, firmware, and configuration together with resilience mechanisms help to protect the system from compromise. Depending on the hardware platform, the hardware integrity, a kernel interface conducting the respective measurements or platform certificates with hardware configuration attributes, is used to maintain the hardware integrity. Hardware integrity refers to the unmodified assembly of hardware to a composite device (i.e., the OC).

As our research is based on DIN VDE V 0831-104 which is a guideline to apply IEC 62443 to the railway domain, we believe that we sufficiently covered all security requirements. For evaluation purposes, these generic security requirements will need to be refined into specific ones that take into account the technology choices made for the implementation of the security architecture. The overall approach is similar to SREP (Security Requirements Engineering Process) [273, 274]: first relevant domain-specific security requirements were determined using standard DIN VDE V 0831-104 (i.e., IEC 62443 applied to the railway domain) [66] and, second, the specific knowledge about the solution's architecture (including related functional limitations and security threats) is used by security experts to elicit system-specific requirements that can later be utilized in the solution's validation and testing.

### 7.6.2. Test Environment

To evaluate the implemented architecture, we integrated the prototype of our implementation into a test-bed for railway operations simulations. The test-bed is based on a digital model railway with currently two switches and two signals as field elements on which a model train is running (cf. Figure 7.3).

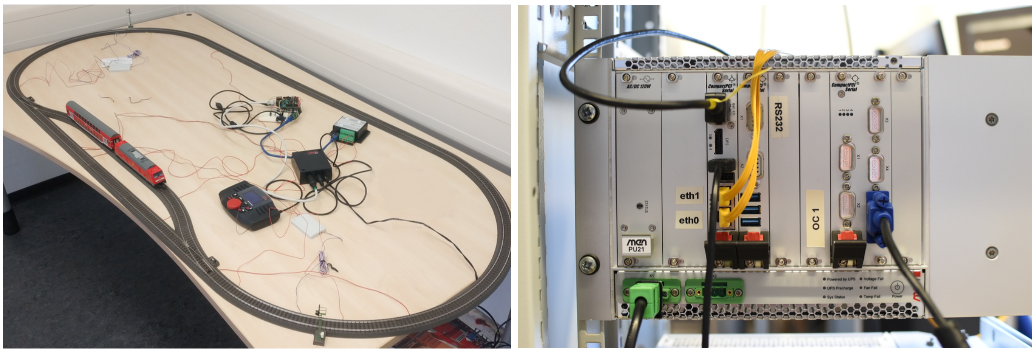


Figure 7.3: Test-Bed for Railway Simulations (MEN G22 on the Right)

The field elements can be controlled by a legacy OC or an OC that implements our security architecture. The hardware is the development board MEN G22 equipped with TPM 2.0. A safety backend implements functions for railway interlocking and operations control and a security backend implements MDM and SOC functionalities providing firmware and configuration updates and performing periodic platform attestations. The communication of the OC to the model railway is handled by Railuino[20]. The test-bed allows simulating local and remote adversaries and play through various test scenarios to evaluate our solution and the effects it has on the railway operations. Examples of test scenarios used to evaluate the above mitigations are as follows:

**Detecting compromised OCs:** A local adversary boots manipulated firmware on an OC to control it as they like. To make this attack scalable a remote adversary manipulates an update image in the MDM.

**Preventing OC's resource exhaustion:** A software stack used to built a (security) application on an OC has a software bug or an exploit crafted by an adversary leading to resources exhaustion and undesired behavior in the OC.

**Detecting injected/spoofed commands:** An adversary manages to hijack the communication between an OC and interlocking and injects/spoofs control messages to make the OC to perform unsafe operations. Alternatively, the adversary may try to cause the denial of service by flooding the OC with messages or by sending specially crafted messages.

### 7.6.3. Integration of IoRT Applications

An interesting aspect of the Internet of Railway Things (IoRT) usage in the command and control domain is that the CCS can be considered both as a target system for services, such as Prediction-based Maintenance (PbM) and Situational Awareness (SE), and as an IoRT component itself. In addition, the IoRT sensors and gateways must meet not only environmental conditions but also comply with strict safety and security requirements to communicate over the railway WAN, which is otherwise used only to ensure railway operations. Another special aspect is that the IoRT components like connected command and control devices in the field element area are available to local adversaries, who can manipulate the systems without being noticed. When using publicly available COTS, such attacks can easily scale. These attack opportunities pose a challenge to the IT security of command and control systems and IoRT services alike.

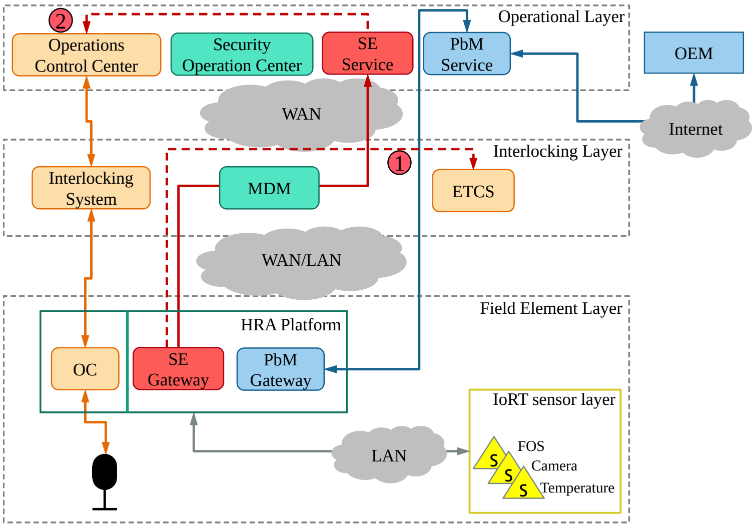


Figure 7.4: Architecture for Secure Integration of IoRT Applications

In Figure 7.4, we show how the IoRT application can be securely deployed on the OC using the proposed security architecture HRA. The basic idea is to treat IoRT applications similar to the additional security applications and leverage the partitioning mechanisms of the chosen SK to ensure their safe integration on the HRA-OC platform. The *SE Gateway* and *PbM Gateway* applications in Figure 7.4 enable the necessary IoRT gateway functions on the OC connecting various

[20] The Railuino is an Arduino Uno Rev3 with an ATmega328P, `https://code.google.com/archive/p/railuino/`

track-side sensors to the respective service backend. A detailed description of the proposed solution together with the usage scenarios and their security considerations can be found in our paper [24].

Currently, the OC can communicate only with the systems at the ILS layer and only via the internal railway WAN. In this case, the safety communication (commands and signals) runs only through ILS, i.e., a direct connection to third-parties or a train driver is not possible. Moreover, an OC cannot be queried directly by the SOC or another service and thus can function as an IoRT gateway only to a limited extent (without Internet or cloud connection). Therefore, the MDM in our solution serves additionally as a security component that forwards both security alerts and IoRT data to the responsible control center, OCC or SOC. With the HRA-based OC, certain edge computing functions are directly supported. For example, the security system can perform local diagnostics and restore a secure state if necessary.

## 7.7. Chapter Summary

Safety and security have been traditionally two different and isolated worlds. Safety certification, especially in the railway sector, does not consider security measures. As a result, one of the greatest challenges of the ongoing railway digitalization, is how to guarantee the transportation safety of the new IT-based railway systems now open to malicious adversaries. In this chapter, we define security requirements using the dedicated security requirements engineering process for railway signaling systems described in the German guideline DIN VDE V 0831-104. We use the derived requirements to design a security architecture for safety-critical railway signaling systems based on a hardware platform allowing to run mixed-criticality applications. The proposed security architecture is based on the MILS concept to comply with the identified security requirements. The concept is implemented using a separation kernel that controls resource sharing between safety and security applications. The security applications of our architecture include health monitoring, authenticated boot, remote attestation procedures, secure updates, and anomaly detection services. These capabilities provide a wide spectrum of security-related information including complementary indicators that improve security assessments as well as cryptographic integrity evidence that can be conveyed to management systems to prove that a device is a trustworthy system. Additionally, secure audit logs are created that are able to document past states a device was in with the highest level of assurance. We analyze how the proposed security architecture fulfills the requirements we derived, design a test-bed for practical evaluations, and look into further application areas like IoRT.

# Part IV.

# Conclusion and Future Work

# 8. Concluding Remarks

Safety-critical infrastructures become more connected and vulnerable to malicious attacks, which may have large-scale impacts and cause serious hazardous situations. Attacks on the Vehicle-to-Grid (V2G) infrastructure can disrupt transportation by preventing Electric Vehicles (EVs) from charging, compromise billing data and privacy of EV users, and even destabilize the energy grid. Similarly devastating are the potential outcomes of malicious actions against railway Command and Control System (CCS) ranging from delays or cancellation of the service to train accidents such as derailments or collisions. Currently deployed systems provide in the best case only basic point-to-point communication security and often rely on outdated, unrealistic threat models.

In this thesis, we investigated security and safety of V2G and railway CCS systems and addressed the aspects of secure communications, platform assurance as well as safety and security co-engineering.

In Part II, we investigated different aspects of the V2G communication security. In Chapter 3, we analyzed V2G communication protocols to identify security gaps with regard to secure service provisioning and developed a new protocol Plug-and-Patch (PnP) that enables secure and reliable remote update of EVs via the V2G infrastructure in the presence of untrusted intermediaries [3]. PnP integrates the Trusted Platform Module (TPM) 2.0 to verify the vehicle's software state before and after the update installation and provide an evidence for the manufacturer. The proposed design builds on the communication standards ISO 15118 and OCPP 2.0 and does not require from charge points to support any proprietary service implementations. Due to the generic nature of our design, PnP can be used to implement any arbitrary vehicle-related Value Added Services (VAS).

In Chapter 4, we explored a fraud scenario related to the consumption of services and showed the possibility of efficient fraud detection based on process specifications that describe benign user behavior instead of data patterns associated with the fraud [5]. We described a new model-based method FCD able to identify individual fraudulent transactions and complete fraud chains. During our evaluation using a synthetic data set, the proposed method showed better recognition performance compared to classical machine learning algorithms. Moreover, the usage of process models allowed for the easy interpretation of alerts helping analysts to report fraud more efficiently.

In Part III, we investigated the problem of the system security in the V2G systems and railway CCSs and safety and security co-engineering approaches. In Chapter 5, we designed the security architectures TrustEV, HIP, and HIP 2.0 to provide the necessary Level of Assurance (LoA) with respect to vehicle's authentication in V2G application scenarios, such as the Plug-and-Charge (PnC) service. All three architectures rely on a certified automotive-class Hardware Security Module (HSM) installed in the vehicle to enable secure credential provisioning. TrustEV protects the identity credentials being delivered and installed in an EV via ISO 15118 under strong adversary scenarios [1]. HIP additionally allows the service provider to outsource the secure generation of sensitive cryptographic credentials to the vehicle itself and ensures that these credentials can be used for the authentication only if the required security properties are met [7]. HIP 2.0 extends these guarantees to include asynchronous provisioning and enrollment methods using the intermediate storage of credentials [8]. We integrated the new security functionality into the protocol flow of ISO 15118 in a backward compatible manner as protocol extensions. We instantiated our solutions using a hardware TPM 2.0 and analyzed the performance trade-offs and the compatibility of the TPM 2.0 interface with the ISO 15118 standard. This way, companies like Volkswagen [132] that are planning to deploy TPMs in their vehicles can use our results to detect potential performance bottlenecks regarding V2G scenarios. Other companies can use our results as a benchmark for an alternative implementation of the proposed security features. The standardization committee responsible for the ISO 15118 standard series[21] accepted our TrustEV security extension, which will be officially released as part of ISO/IEC 15118-20. The discussion about integrating the HIP and HIP 2.0 features was deferred to the next revision.

In Chapter 6, we investigated safety and security co-engineering and applied the integrity assurance as a safety measure. We developed the security architecture `secEVCS` aiming to prevent hazardous situations during charging by allowing only vehicles with manufacturer-approved charging systems to (dis)charge electric energy at charge points. This guarantee is achieved via components binding based on security policy assertions restricting the system's operation

[21] ISO/TC 22/SC 31/JWG 1 "Joint ISO/TC 22/SC 31 – IEC/TC 69 WG: Vehicle to grid communication interface (V2G CI)"

to the safe configuration. We evaluated our solution using a hardware TPM 2.0 and ISO 15118 and analyzed the performance bottlenecks and the required optimizations for the compatible operation [2].

In Chapter 7, we investigated safety and security co-engineering for the safety-critical railway CCS, which posed an opposite challenge. We designed a security architecture HRA for a digital Object Controller (OC) used in railway signaling that enables the integration of additional security functions on the hardware platform hosting the safety application without voiding its railway safety certifications [11, 10, 9]. This architecture leverages the properties of the Separation Kernel (SK) and Multiple Independent Levels of Security (MILS) concepts to ensure the freedom of interference between the two domains.

Overall, we explored security issues of the communication infrastructures emerging in the area of e-mobility and digital railways to provide connectivity to safety-critical systems and proposed solutions using open security and communication standards to mitigate them. We addressed aspects of secure end-to-end communications under weak trust assumptions and system hardening techniques ensuring both security and safety properties to increase resilience of safety-critical systems to malicious attacks. The proposed solutions make provisions for the compatibility with the major domain-specific standards in e-mobility and digital railways to facilitate their deployments in today's infrastructures and facilitate the secure transformation.

Our work opens several interesting topics with regard to security and safety in e-mobility and digital railways worthwhile to investigate further. For V2G communications, the main challenge remains to develop a comprehensive security framework that provides end-to-end guarantees in the presence of untrusted intermediaries and integrate it into the existing ecosystem. With the current patchwork of service providers and communication protocols, neither fair access to EV charging nor control over the transferred information can be assured. Secure EV identities proposed in this thesis for ISO 15118's PnC is one of the first steps in this direction.

Further research on extending automotive HSMs to support the security features identified for e-mobility use cases is important to enable a unified assurance level across car manufacturers, those supporting TPM 2.0 and those opting for more flexible solutions. It is also important to sensitize the Trusted Computing Group (TCG) for the requirements posed by ISO 15118-20 to make the specification compatible with this upcoming edition.

One of the challenges for the large-scale deployment of ISO 15118 and secure communications in the V2G ecosystem in general is the availability of the V2G Public Key Infrastructure (PKI). Therefore, it would be interesting to investigate approaches for decentralized trust management in the V2G infrastructure without trusted third parties.

Moreover, since attacks and manipulated components can never be completely ruled out, mechanisms are needed to assess the trustworthiness of (unknown) components and to take measures to ensure that the overall system remains functional even if some of the components are compromised. Health monitoring and recovery approaches are necessary to ensure the sustainability of the critical infrastructures in scope.

Since charging and discharging processes involve transfer of funds, they can provide an incentive for financial fraud. Service providers are already monitoring transactions performed with a vehicle's ID but the possible fraud scenarios are not well understood yet. Possible examples could be tax evasion, or violation of the contract conditions. Thus, the development of fraud detection systems that analyze data in various provider backends is a promising research direction.

# List of Abbreviations

| | | |
|---|---|---|
| **AIK** | Attestation Identity Key | 30 |
| **API** | Application Programming Interface | 62 |
| **BMS** | Battery Management System | 97 |
| **CA** | Certificate Authority | 56 |
| **CCH** | Contract Clearing House | 10 |
| **CDR** | Charge Detail Record | 58 |
| **CPID** | Charge Point Identifier | 11 |
| **CPO** | Charge Point Operator | 53 |
| **CPS** | Certificate Provisioning Service | 56 |
| **CRL** | Certificate Revocation List | 81 |
| **CS** | Central System | 14 |
| **CSMS** | Charging Station Management System | 14 |
| **DAA** | Direct Anonymous Attestation | 94 |
| **DER** | Distributed Energy Resource | 9 |
| **DIS** | Draft International Standard | 55 |
| **DoS** | Denial of Service | 108 |
| **DSO** | Distribution System Operator | 53 |
| **EA** | Enhanced Authorization | 20 |
| **ECU** | Electronic Control Unit | 102 |
| **EC** | Elliptic Curve | 67 |
| **ECC** | Elliptic Curve Cryptography | 60 |
| **ECDH** | Elliptic Curve Diffie-Hellman | 57 |
| **EIM** | External Identification Means | 56 |
| **EMAID** | E-Mobility Account Identifier | 56 |
| **eMIP** | eMobility Protocol Inter-Operation | 17 |
| **EV** | Electric Vehicle | 127 |
| **EVCC** | Electric Vehicle Communication Controller | 97 |
| **EVCCID** | Electric Vehicle Communication Controller Identifier | 12 |
| **EVSE** | Electric Vehicle Supply Equipment | 53 |
| **EVSEID** | Electric Vehicle Supply Equipment Identifier | 12 |
| **FDIS** | Final Draft International Standard | 55 |
| **GDPR** | General Data Protection Regulation | 15 |
| **HLC** | High Level Communication | 9 |
| **HSM** | Hardware Security Module | 127 |
| **HMAC** | Hash Message Authentication Code | 20 |

| | | |
|---|---|---|
| **ID** | Identifier | 100 |
| **IP** | Intellectual Property | 26 |
| **JSON** | Javascript Object Notation | 14 |
| **JWG** | Joint Working Group | 10 |
| **TC** | Technical Committee | 10 |
| **SC** | Sub Committee | 10 |
| **OSI** | Open Systems Interconnection | 10 |
| **KDF** | Key Derivation Function | 101 |
| **KE** | Key Exchange | 31 |
| **MAC** | Message Authentication Code | 28 |
| **MitM** | Man-in-the-Middle | 28 |
| **eMSP** | E-Mobility Service Provider | 53 |
| **EMS** | Energy Management System | 16 |
| **OCA** | Open Charge Alliance | 13 |
| **OCHP** | Open Clearing House Protocol | 83 |
| **OCPI** | Open Charge Point Interface | 83 |
| **OCPP** | Open Charge Point Protocol | 53 |
| **OCSP** | Online Certificate Status Protocol | 81 |
| **OEM** | Original Equipment Manufacturer | 98 |
| **OICP** | Open Intercharge Protocol | 83 |
| **PCID** | Provisioning Certificate Identifier | 56 |
| **PII** | Personally Identifiable Information | 15 |
| **PKI** | Public Key Infrastructure | 128 |
| **PnC** | Plug-and-Charge | 127 |
| **PnP** | Plug-and-Patch | 127 |
| **PQ** | Post Quantum | 60 |
| **RAM** | Random Access Memory | 59 |
| **SA** | Secondary Actor | 56 |
| **SECC** | Supply Equipment Communication Controller | 98 |
| **SOAP** | Simple Object Access Protocol | 14 |
| **SOTA** | Software-Over-the-Air | 26 |
| **OTA** | Over-the-Air | 15 |
| **TCG** | Trusted Computing Group | 128 |
| **TLS** | Transport Layer Security | 54 |
| **TPM** | Trusted Platform Module | 127 |
| **VAS** | Value Added Services | 127 |
| **V2V** | Vehicle-to-Vehicle | 37 |
| **VUC** | Vehicle Update Certificate | 30 |
| **OUC** | OEM Update Certificate | 30 |
| **V2G** | Vehicle-to-Grid | 127 |
| **WAN** | Wide Area Network | 108 |

| | | |
|---|---|---|
| **XSD** | XML Schema Definition | 91 |
| **APA** | Asynchronous Product Automata | 41 |
| **IT** | Information Technology | 107 |
| **MMT** | Mobile Money Transfer | 133 |
| **PSA** | Predictive Security Analyzer | 38 |
| **RG** | Reachability Graph | 41 |
| **ML** | Money Laundering | 133 |
| **MNO** | Mobile Network Operator | 37 |
| **PSA@R** | Predictive Security Analysis at Runtime | 37 |
| **AML** | Anti-Money Laundering | 37 |
| **MD** | Money Deposit | 38 |
| **MW** | Money Withdrawal | 38 |
| **AR** | Airtime Recharge | 38 |
| **MP** | Merchant Payment | 38 |
| **C2C** | Client-to-Client transfer | 38 |
| **CAN** | Controller Area Network | 98 |
| **DICE** | Device Identifier Composition Engine | 97 |
| **DSO** | Distribution System Operator | 53 |
| **EVCS** | Electric Vehicle Charging System | 97 |
| **KDF** | Key Derivation Function | 101 |
| **NV** | Non-Volatile | 101 |
| **VIN** | Vehicle Identification Number | 98 |
| **MTM** | Mobile Trusted Module | 94 |
| **SSH** | Secure Shell | 103 |
| **SHE** | Secure Hardware Extensions | 54 |
| **V2X** | Vehicle-to-Everything | 94 |
| **PCR** | Platform Configuration Register | 118 |
| **TLS** | Transport Layer Security | 54 |
| **SRK** | Storage Root Key | 62 |
| **PoC** | Proof-of-Concept | 55 |
| **CSR** | Certificate Signing Request | 55 |
| **GCS** | Global Certificate Store | 55 |
| **SoC** | System on Chip | 94 |
| **DAA** | Direct Anonymous Attestation | 94 |
| **LoA** | Level of Assurance | 127 |
| **DB** | Deutsche Bahn | 107 |
| **TDS** | Train Detection System | 108 |
| **EU** | European Union | 107 |
| **HRA** | Haselnuss Reference Architecture | 117 |
| **SOC** | Security Operations Center | 107 |
| **COTS** | Commercial of the Shelf | 107 |

| | | |
|---|---|---|
| **OCC** | Operations Control Center | 107 |
| **RaSTA** | Rail Safe Transport Application | 108 |
| **SIL** | Safety Integrity Level | 7 |
| **IoRT** | Internet of Railway Things | 122 |
| **BMBF** | Federal Ministry of Education and Research | 8 |
| **SL** | Security Level | 114 |
| **ILS** | Interlocking System | 107 |
| **PSL** | Preliminary Security Level | 114 |
| **FR** | Foundational Requirement | 114 |
| **MDM** | Maintenance and Data Management System | 107 |
| **OC** | Object Controller | 128 |
| **CI** | Critical Infrastructure | 119 |
| **MILS** | Multiple Independent Levels of Security | 128 |
| **SK** | Separation Kernel | 128 |
| **CC** | Common Criteria | 54 |
| **ROT** | Root of Trust | 118 |
| **TUDA** | Time-Based Uni-Directional Attestation | 118 |
| **TSS** | TPM Software Stack | 118 |
| **CCS** | Command and Control System | 127 |
| **IDS** | Intrusion Detection System | 119 |
| **IACS** | Industrial Automation and Control System | 116 |
| **CPU** | Central Processing Unit | 118 |
| **MMU** | Memory Management Unit | 118 |
| **PbM** | Prediction-based Maintenance | 122 |
| **SE** | Situational Awareness | 122 |

# List of Figures

# List of Publications

[1] Andreas Fuchs, Dustin Kern, Christoph Krauß, and Maria Zhdanova. “TrustEV: Trustworthy Electric Vehicle Charging and Billing”. In: *SAC’20: The 35th ACM/SIGAPP Symposium on Applied Computing*. Ed. by Chih-Cheng Hung, Tomás Cerný, Dongwan Shin, and Alessio Bechini. ACM, Mar. 2020, pp. 1706–1715.

[2] Andreas Fuchs, Dustin Kern, Christoph Krauß, and Maria Zhdanova. “Securing Electric Vehicle Charging Systems Through Component Binding”. In: *Computer Safety, Reliability, and Security - 39th International Conference, SAFECOMP 2020*. Ed. by António Casimiro, Frank Ortmeier, Friedemann Bitsch, and Pedro Ferreira. Vol. 12234. Lecture Notes in Computer Science. Springer, Sept. 2020, pp. 387–401.

[3] Lucas Buschlinger, Markus Springer, and Maria Zhdanova. “Plug-and-Patch: Secure Value Added Services for Electric Vehicle Charging”. In: *ARES 2019: The 14th International Conference on Availability, Reliability and Security*. ACM, Aug. 2019, 2:1–2:10.

[4] Daniel Zelle, Markus Springer, Maria Zhdanova, and Christoph Krauß. “Anonymous Charging and Billing of Electric Vehicles”. In: *ARES 2018: The 13th International Conference on Availability, Reliability and Security*. Ed. by Sebastian Doerr, Mathias Fischer, Sebastian Schrittwieser, and Dominik Herrmann. ACM, Aug. 2018, 22:1–22:10.

[5] Maria Zhdanova, Jürgen Repp, Roland Rieke, Chrystel Gaber, and Baptiste Hemery. “No Smurfs: Revealing Fraud Chains in Mobile Money Transfers”. In: *ARES 2014: Ninth International Conference on Availability, Reliability and Security*. IEEE Computer Society, Sept. 2014, pp. 11–20.

[6] Steffen Schulz, Ahmad-Reza Sadeghi, Maria Zhdanova, Hossen A. Mustafa, Wenyuan Xu, and Vijay Varadharajan. “Tetherway: a Framework for Tethering Camouflage”. In: *WISEC 2012: ACM Conference on Security and Privacy in Wireless and Mobile Networks*. Ed. by Marwan Krunz, Loukas Lazos, Roberto Di Pietro, and Wade Trappe. ACM, Apr. 2012, pp. 149–160.

[7] Andreas Fuchs, Dustin Kern, Christoph Krauß, and Maria Zhdanova. “HIP: HSM-based Identities for Plug-and-Charge”. In: *ARES 2020: The 15th International Conference on Availability, Reliability and Security*. Ed. by Melanie Volkamer and Christian Wressnegger. ACM, Aug. 2020, 33:1–33:6.

[8] Andreas Fuchs, Dustin Kern, Christoph Krauß, Maria Zhdanova, and Ronald Heddergott. “HIP20: Integration of Vehicle-HSM-Generated Credentials into Plug-and-Charge Infrastructure”. In: *4th ACM Computer Science in Cars Symposium*. ACM, Dec. 2020.

[9] Michael Eckel, Don Kuzhiyelil, Christoph Krauß, Maria Zhdanova, Stefan Katzenbeisser, Jasmin Cosic, Matthias Drodt, and Jean-Jacques Pitrolle. “Implementing a Security Architecture for Safety-Critical Railway Infrastructure”. In: *IEEE International Symposium on Secure and Private Execution Environment Design (SEED)* (2021).

[10] Markus Heinrich, Tsvetoslava Vateva-Gurova, Tolga Arul, Stefan Katzenbeisser, Neeraj Suri, Henk Birkholz, Andreas Fuchs, Christoph Krauß, Maria Zhdanova, Don Kuzhiyelil, Sergey Tverdyshev, and Christian Schlehuber. “Security Requirements Engineering in Safety-Critical Railway Signalling Networks”. In: *Security and Communication Networks* 2019 (), 8348925:1–8348925:14.

[11] Henk Birkholz, Christoph Krauß, Maria Zhdanova, Don Kuzhiyelil, Tolga Arul, Markus Heinrich, Stefan Katzenbeisser, Neeraj Suri, Tsvetoslava Vateva-Gurova, and Christian Schlehuber. “A Reference Architecture for Integrating Safety and Security Applications on Railway Command and Control Systems”. In: *International Workshop on MILS: Architecture and Assurance for Secure Systems, MILS@DSN 2018*. Ed. by Sergey Tverdyshev. June 2018.

[12] Roland Rieke, Maria Zhdanova, and Jürgen Repp. “Security Compliance Tracking of Processes in Networked Cooperating Systems”. In: *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)* 6.2 (2015), pp. 21–40.

[13] Roland Rieke, Maria Zhdanova, and Jürgen Repp. “Security and Business Situational Awareness”. In: *Cyber Security and Privacy – 4th Cyber Security and Privacy Innovation Forum, CSP Innovation Forum 2015, Revised Selected Papers*. Ed. by Frances Cleary and Massimo Felici. Vol. 530. Communications in Computer and Information Science. Springer, Apr. 2015, pp. 103–115.

[14] Patrick Koeberl, Vinay Phegade, Anand Rajan, Thomas Schneider, Steffen Schulz, and Maria Zhdanova. “Time to Rethink: Trust Brokerage Using Trusted Execution Environments”. In: *Trust and Trustworthy Computing - 8th International Conference, TRUST 2015*. Ed. by Mauro Conti, Matthias Schunter, and Ioannis G. Askoxylakis. Vol. 9229. Lecture Notes in Computer Science. Springer, Aug. 2015, pp. 181–190.

[15] Roland Rieke, Jürgen Repp, Maria Zhdanova, and Jörn Eichler. “Monitoring Security Compliance of Critical Processes”. In: *22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2014*. IEEE Computer Society, Feb. 2014, pp. 552–560.

[16] Roland Rieke, Maria Zhdanova, Jürgen Repp, Romain Giot, and Chrystel Gaber. “Verhaltensanalyse zur Erkennung von Missbrauch mobiler Geldtransferdienste”. In: *Sicherheit 2014: Sicherheit, Schutz und Zuverlässigkeit, Beiträge der 7. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)*. Ed. by Stefan Katzenbeisser, Volkmar Lotz, and Edgar R. Weippl. Vol. P-228. LNI. GI, Mar. 2014, pp. 271–282.

[17] Roland Rieke, Maria Zhdanova, Jürgen Repp, Romain Giot, and Chrystel Gaber. “Fraud Detection in Mobile Payments Utilizing Process Behavior Analysis”. In: *ARES 2013: International Conference on Availability, Reliability and Security*. IEEE Computer Society, Sept. 2013, pp. 662–669.

[18] Maria Zhdanova, Dustin Kern, and Christoph Krauß. *TPM-protected EVCC Credentials for ISO 15118-20. Proposal for Changes in ISO 15118-20 preFDIS*. Tech. rep. Fraunhofer SIT.

[19] Dustin Kern, Christoph Krauß, and Maria Zhdanova. *System Security Mechanisms for Electric Vehicles and Charge Points Supporting ISO 15118 – Proposal for a Technical Guideline*. SIT-TR-2019-04. Fraunhofer SIT.

[20] Maria Zhdanova and Christoph Krauß. *Secure Electric Vehicle Communication Controller for ISO 15118 – Proposal for a Protection Profile*. SIT-TR-2019-01. Fraunhofer SIT.

[21] Maria Zhdanova. *Secure Charge Point Communication Controller for ISO 15118 and Backend Connection – Proposal for a Protection Profile*. SIT-TR-2019-02. Fraunhofer SIT.

[22] Daniel Zelle, Maria Zhdanova, Christoph Krauß, Christoph Leicht, Michael Blaž, Roberto Cuerdo, Jannes Langemann, Simone Mühe, Daniel Stetter, Marc Schmid, Tobias Höpfer, Marco Mittelsdorf, Lukas Smoluch, and Christian Seipel. *Gegenüberstellung von Ladeeinrichtungsarchitekturentwürfen für eichrechtskonforme authentifizierte Ladevorgänge unter Einbeziehung eines SMGWs*. Tech. rep. Dec. 2021.

[23] Jianzhong Huang, Sanat Sarda, Michael Kasper, Bi Xin, Wang Yi Estelle, Maria Zhdanova, and Christoph Krauß. *China Electric Vehicle and Connected Vehicle Security And Privacy. Government, Industry and Standardization Perspective (2015-2021)*. Tech. rep. Fraunhofer SIT, 2021.

[24] Tolga Arul, Jasmin Cosic, Matthias Drodt, Marcus Friedrich, Markus Heinrich, Michael Kant, Stefan Katzenbeisser, Helmut Klarer, Patrick Rauscher, Max Schubert, Gerhard Still, Detlef Wallenhorst, and Maria Zhdanova. *IT/OT-Security for Internet of Railway Things (IoRT)*. Tech. rep. Deutsche Bahn RailLab. Working Group (WG) CYSIS.

[25] Christoph Krauß, Maria Zhdanova, Michael Eckel, Stefan Katzenbeisser, Markus Heinrich, Don Kuzhiyelil, Jasmin Cosic, and Matthias Drodt. “Projekt HASELNUSS - IT-Sicherheitsarchitektur für die nächste Generation der Leit- und Sicherheitstechnik”. In: *Deine Bahn* 12 (Dec. 2020), pp. 57–61.

[26] Christoph Krauß, Maria Zhdanova, Michael Eckel, Stefan Katzenbeisser, Markus Heinrich, Don Kuzhiyelil, Jasmin Cosic, and Matthias Drodt. “A Security Architecture for Protecting Safety-Critical Railway Infrastructure”. In: *Embedded World Conference 2021 Digital Proceedings*. embedded world conference 2021. Nürnberg, Germany: weka fachmedien GmbH, 2021. ISBN: 9783645501897.

[27] Frank Brosi, Andreas Harner, Christoph Krauß, Christian Seipel, Markus Springer, Michael Staubermann, André Suhr, Stephan Voit, and Maria Zhdanova. *Handlungsempfehlungen des Förderprojekts DELTA an die Politik, Normung und Standardisierung, Ladesäulenhersteller/-betreiber und Automobilhersteller, und zukünftige Förderprojekte*. Tech. rep. DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE, 2019.

[28] Maria Zhdanova, Christoph Krauß, and Dustin Kern. *Entwicklung und prototypische Umsetzung von Schutzprofil und Richtlinien zum Laden und Abrechnen in der Elektromobilität*. Abschlussbericht 01MX15014E. Verbundprojekt DELTA: Datensicherheit und -integrität in der Elektromobilität beim Laden und eichrechtskonformen Abrechnen, 2019.

[29] Christoph Krauß, Michael Eckel, and Maria Zhdanova. *Gerätesicherheit und Health Monitoring für hardwarebasierte Sicherheitsplattform für Eisenbahn Leit- und Sicherungstechnik*. Abschlussbericht 16KIS0432K. Verbundprojekt HASELNUSS: Hardwarebasierte Sicherheitsplattform für Eisenbahn Leit- und Sicherungstechnik, 2021.

[30] Jürgen Repp, Maria Zhdanova, and Roland Rieke. *Methods and tools for reasoning about security with uncertain knowledge*. Deliverable D4.1.3. FP7-257475 MASSIF European project, 2012.

[31] Jürgen Repp, Roland Rieke, Rodrigo Diaz, and Maria Zhdanova. *Predictive Security Analyser*. Deliverable D4.2.3. FP7-257475 MASSIF European project, 2013.

[32] Roland Rieke, Jürgen Repp, and Maria Zhdanova. *Process Model and Dynamic Simulation and Analysis Modelling Framework*. Deliverable D4.2.2. FP7-257475 MASSIF European project, Oct. 2012.

[33] Maria Zhdanova, Jürgen Repp, and Roland Rieke. *Multi-level Abstraction Concept*. Deliverable D4.1.2. FP7-257475 MASSIF European project, Oct. 2012.

[34] Paulo Verissimo et al. *MASSIF Architecture Document*. Tech. rep. FP7-257475 MASSIF European project, Apr. 2012.

# Bibliography

[35] *Global EV Outlook 2021*. 2020. URL: www.iea.org/reports/global-ev-outlook-2021/trends-and-developments-in-electric-vehicle-markets.

[36] Kristien Clement-Nyns, Edwin Haesen, and Johan Driesen. “The impact of vehicle-to-grid on the distribution grid”. In: *Electric Power Systems Research* 81.1 (2011), pp. 185–192. ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2010.08.007.

[37] *ISO/IEC 15118-1. Road vehicles – Vehicle to grid communication interface – Part 1: General information and use-case definition*. International Standard 15118-1:2013. Geneva, Switzerland: International Organization for Standardization and International Electrotechnical Commission, Apr. 2013.

[38] *ISO/IEC 15118-2. Road vehicles – Vehicle-to-Grid Communication Interface – Part 2: Network and application protocol requirements*. International Standard 15118-2:2014. Edition 1. International Organization for Standardization and International Electrotechnical Commission, 2014.

[39] *ISO/IEC FDIS 15118-20. Road vehicles – Vehicle-to-Grid Communication Interface – Part 20: Network and application protocol requirements*. Final Draft International Standard FDIS 15118-20. Geneva, Switzerland: International Organization for Standardization and International Electrotechnical Commission, 2021.

[40] P. Rademakers and P. Klapwijk. *EV Related Protocol Study*. Jan. 2017. URL: https://www.elaad.nl/uploads/downloads/downloads_download/EV_related_protocol_study_v1.1.pdf.

[41] Mustafa A. Mustafa, Ning Zhang, Georgios Kalogridis, and Zhong Fan. “Smart electric vehicle charging: Security analysis”. In: *IEEE PES Innovative Smart Grid Technologies Conference (ISGT 2013)*. 2013, pp. 1–6. DOI: 10.1109/ISGT.2013.6497830.

[42] Peiyi Sun, Roeland Bisschop, Huichang Niu, and Xinyan Huang. “A Review of Battery Fires in Electric Vehicles”. In: *Fire Technology* (2020). ISSN: 1572-8099.

[43] Anthony Bahadir Lopez, Korosh Vatanparvar, Atul Prasad Deb Nath, Shuo Yang, Swarup Bhunia, and Mohammad Abdullah Al Faruque. “A Security Perspective on Battery Systems of the Internet of Things”. In: *Journal of Hardware and Systems Security* (2017). ISSN: 2509-3436.

[44] James G. Wright and Stephen D. Wolthusen. “Access Control and Availability Vulnerabilities in the ISO/IEC 61850 Substation Automation Protocol”. In: *Critical Information Infrastructures Security*. Ed. by Grigore Havarneanu, Roberto Setola, Hypatia Nassopoulos, and Stephen Wolthusen. Cham: Springer International Publishing, 2017, pp. 239–251. ISBN: 978-3-319-71368-7.

[45] Adrian Dabrowski, Johanna Ullrich, and Edgar R. Weippl. “Grid Shock: Coordinated Load-Changing Attacks on Power Grids: The Non-Smart Power Grid is Vulnerable to Cyber Attacks as Well”. In: *Proceedings of the 33rd Annual Computer Security Applications Conference*. ACSAC 2017. Orlando, FL, USA: Association for Computing Machinery, 2017, pp. 303–314. ISBN: 9781450353458. DOI: 10.1145/3134600.3134639.

[46] Saleh Soltan, Prateek Mittal, and H. Vincent Poor. “BlackIoT: IoT Botnet of High Wattage Devices Can Disrupt the Power Grid”. In: *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 15–32. ISBN: 978-1-939133-04-5.

[47] Richard Baker and Ivan Martinovic. “Losing the Car Keys: Wireless PHY-Layer Insecurity in EV Charging”. In: *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 407–424. ISBN: 978-1-939133-06-9.

[48] R. Falk and S. Fries. “Electric vehicle charging infrastructure security considerations and approaches”. In: *INTERNET 2012: The Fourth International Conference on Evolving Internet* (2012), pp. 58–64.

[49] Sébastien Dudek, Jean-Christophe Delaunay, and Vincent Fargues. *V2G Injector: Whispering to cars and charging units through the Power-Line*. Tech. rep. SSTIC 2019, 2019.

[50] K. Bao, H. Valev, M. Wagner, and H. Schmeck. “A threat analysis of the vehicle-to-grid charging protocol ISO 15118”. In: *Computer Science - Research and Development* (Sept. 2017). ISSN: 1865-2042. DOI: 10.1007/s00450-017-0342-y.

[51] S. Lee, Y. Park, H. Lim, and T. Shon. “Study on Analysis of Security Vulnerabilities and Countermeasures in ISO/IEC 15118 based Electric Vehicle Charging Technology”. In: *2014 International Conference on IT Convergence and Security (ICITCS)*. Oct. 2014, pp. 1–4.

[52] H. Chaudhry and T. Bohn. “Security concerns of a plug-in vehicle”. In: *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*. Jan. 2012, pp. 1–6.

[53] Kim Zetter. *Sony Got Hacked Hard: What We Know and Don’t Know So Far*. 2014. URL: https://www.wired.com/2014/12/sony-hack-what-we-know/.

[54] Dennis Fisher. *DigiNotar Says Its CA Infrastructure Was Compromised*. 2011. URL: https://threatpost.com/diginotar-says-its-ca-infrastructure-was-compromised-083011/75594/.

[55] Robert McMillan. *Comodo Hacker Claims Another Certificate Authority*. 2011. URL: https://www.pcworld.com/article/223760/article.html.

[56] Mathias Dalheimer. *Chaos Computer Club hacks e-motor charging stations*. Dec. 2017. URL: https://www.ccc.de/en/updates/2017/e-motor.

[57] Huajiang Chen and Wu Ming. *Remotely Rooting Charging Station For Fun And Maybe Profit*. Aug. 2021. URL: http://chv.link/kevin2600.

[58] Vangelis Stykas. *Smart car chargers. Plug-n-Play for hackers?* July 2021. URL: https://www.pentestpartners.com/security-blog/smart-car-chargers-plug-n-play-for-hackers/.

[59] C. Miller and C. Valasek. “A Survey of Remote Automotive Attack Surfaces”. In: *Blackhat*. 2014.

[60] Sen Nie, Ling Liu, Yuefeng Du, and Wenkai Zhang. *Over-The-Air: How We Remotely Compromised The Gateway, BCM, and Autopilot ECUs Of Tesla Cars*. Tech. rep. Keen Security Lab of Tencent, 2019.

[61] Ralf-Philipp Weinmann and Benedikt Schmotzle. *TBONE – A zero-click exploit for Tesla MCUs*. Tech. rep. Comsecuris UG, 2021.

[62] *Experimental Security Assessment of BMW Cars: A Summary Report*. Tech. rep. Tencent Keen Security Lab, 2018.

[63] Hans Leister. “ETCS und digitale Technologie für Stellwerke”. In: *Eisenbahn-Revue International* 8-9 (2017).

[64] Christian Schlehuber, Markus Heinrich, Tsvetoslava Vateva-Gurova, Stefan Katzenbeisser, and Neeraj Suri. “Challenges and Approaches in Securing Safety-Relevant Railway Signalling”. In: *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE. 2017, pp. 139–145. DOI: 10.1109/eurospw.2017.63.

[65] Chrystel Gaber, Baptiste Hemery, Mohammed Achemlal, Marc Pasquet, and Pascal Urien. “Synthetic logs generator for fraud detection in mobile transfer services”. In: *International Conference on Collaboration Technologies and Systems*. 2013.

[66] *DIN VDE V 0831-104. Elektrische Bahn-Signalanlagen - Teil 104: Leitfaden für die IT-Sicherheit auf Grundlage der IEC 62443*. Tech. rep. DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE, Oct. 2015.

[67] *PD CLC/TS 50701 Railway applications - Cybersecurity*. Tech. rep. CENELEC, 2020.

[68] Minho Shin, Hwimin Kim, Hyoseop Kim, and Hyuksoo Jang. “Building an Interoperability Test System for Electric Vehicle Chargers Based on ISO/IEC 15118 and IEC 61850 Standards”. In: *Applied Sciences* 6 (May 2016), p. 165. DOI: 10.3390/app6060165.

[69] Myriam Neaimeh and Peter Andersen. “Mind the gap - open communication protocols for vehicle grid integration”. In: *Energy Informatics* 3 (Feb. 2020). DOI: 10.1186/s42162-020-0103-1.

[70] Wolfgang Klebsch, Walter Daumann und Christian Sczyslo, Max Halbritter, and Katharina Vera Boesche. *Studie Ad-hoc-Laden und spontanes Bezahlen*. Tech. rep. VDE Verband der Elektrotechnik Elektronik Informationstechnik e.V. (VDE), 2017. URL: https://www.digitale-technologien.de/DT/Redaktion/DE/Downloads/Publikation/IKT-EM/ikt3-OVAL%5C%20Studie.pdf.

[71] M.A. López, Sebastian Martin, Jose Aguado, and Sebastián de la Torre. “V2G strategies for congestion management in microgrids with high penetration of electric vehicles”. In: *Electric Power Systems Research* 104 (Nov. 2013), pp. 28–34. DOI: 10.1016/j.epsr.2013.06.005.

[72] Ingo Stadler. “Power grid balancing of energy systems with high renewable energy penetration by demand response”. In: *Utilities Policy* 16.2 (2008). Sustainable Energy and Transportation Systems, pp. 90–98. ISSN: 0957-1787. DOI: https://doi.org/10.1016/j.jup.2007.11.006.

[73] C. Höfer, J. Petit, R. Schmidt, and F. Kargl. “POPCORN: privacy-preserving charging for eMobility”. In: *Proceedings of the 2013 ACM workshop on Security, privacy & dependability for cyber vehicles*. ACM. 2013, pp. 37–48.

[74] *EV Charging Systems – Security Architecture*. Tech. rep. European Network for Cyber Security, Apr. 2016.

[75] Benedikt Kirpes, Philipp Danner, Robert Basmadjian, Hermann de Meer, and Christian Becker. “E-Mobility Systems Architecture: a model-based framework for managing complexity and interoperability”. In: *Energy Informatics* 2 (2019), pp. 1–31.

[76] *DIN EN 63110-1:2019-02. Protocol for Management of Electric Vehicles charging and discharging infrastructures – Part 1: Basic Definitions, Use Cases and architectures (IEC 69/612/CD:2018)*. Tech. rep. Draft. International Electrotechnical Commission (IEC), Jan. 2019. URL: https://www.beuth.de/en/draft-standard/din-en-63110-1/299720125.

[77] Charging Interface Initiative e.V. *CharIN e.V. supports ISO/IEC 15118*. Nov. 2017. URL: http://www.charinev.org/fileadmin/Downloads/Papers_and_Regulations/CharIN_Support_of_ISOIEC_15118.pdf.

[78] *Handling of certificates for electric vehicles, charging infrastructure and backend systems within the framework of ISO 15118*. VDE-AR-E 2802-100-1:2019-12. VDE Verband der Elektrotechnik Elektronik Informationstechnik e. V. (VDE), Dec. 2019.

[79] Paul Bertrand. *IEC 63110: Management of EV charging and discharging infrastructure*. Jan. 2020. URL: www.ncl.ac.uk/media/wwwnclacuk/cesi/files/20200115_Meet%5C%20IEC%5C%2063110%5C_%5C%20Paul%5C%20Bertand%5C%20SmartFuture-min.pdf.

[80] *Open Charge Point Protocol (OCPP) 1.6*. Edition 2. Open Charge Alliance. Sept. 2017.

[81] *Open Charge Point Protocol SOAP 1.6*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, Sept. 2017.

[82] *Open Charge Point Protocol JSON 1.6*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, Sept. 2017.

[83] *Open Charge Point Protocol 2.0 - Part 0 - Introduction*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, Apr. 2018.

[84] *Open Charge Point Protocol 2.0 - Part 4 - JSON over WebSockets implementation guide*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, 2018.

[85] *Open Charge Point Protocol 2.0.1 - Part 0 - Introduction*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, 2020.

[86] *Open Charge Point Protocol 2.0.1 - Part 4 - Implementation Guide JSON*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, 2020.

[87] Binod Vaidya and Hussein T. Mouftah. “Deployment of Secure EV Charging System Using Open Charge Point Protocol”. In: *14th International Wireless Communications & Mobile Computing Conference (IWCMC 2018)*. 2018, pp. 922–927. DOI: 10.1109/IWCMC.2018.8450489.

[88] VECTOR. *vCharM: Enterprise-Grade Charging Station Management System*. 2021.

[89] Julian Timpner and Lars Wolf. “Design and Evaluation of Charging Station Scheduling Strategies for Electric Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 15.2 (2014), pp. 579–588. DOI: 10.1109/TITS.2013.2283805.

[90] Marc Pape, Marc Trageser, Chris Martin Vertgewall, and Philipp Goergens. “Flexible Management of Charging Points for E-mobility”. In: *6th IEEE International Energy Conference (ENERGYCon)*. 2020, pp. 876–881. DOI: 10.1109/ENERGYCon48941.2020.9236576.

[91] Oula Lehtinen, Sami Pitkäniemi, Atte Weckman, Mikael Aikio, Michel Mabano, and Matti Lehtonen. “Electric Vehicle Charging Loads in Residential Areas of Apartment Houses”. In: *21st International Scientific Conference on Electric Power Engineering (EPE)*. 2020, pp. 1–6. DOI: 10.1109/EPE51172.2020.9269191.

[92] Eric Rescorla and Tim Dierks. *The Transport Layer Security (TLS) Protocol Version 1.2*. RFC 5246. Aug. 2008. DOI: 10.17487/RFC5246.

[93] John Franks, Phillip Hallam-Baker, Lawrence C. Stewart, Jeffery L. Hostetler, Scott Lawrence, Paul J. Leach, and Ari Luotonen. *HTTP Authentication: Basic and Digest Access Authentication*. RFC 2617. June 1999. DOI: 10.17487/RFC2617.

[94] C. Alcaraz, J. Lopez, and S. Wolthusen. "OCPP Protocol: Security Threats and Challenges". In: *IEEE Transactions on Smart Grid* 8.5 (Sept. 2017), pp. 2452–2459. ISSN: 1949-3053. DOI: 10.1109/TSG.2017.2669647.

[95] J. E. Rubio, C. Alcaraz, and J. Lopez. "Addressing Security in OCPP: Protection Against Man-in-the-Middle Attacks". In: *9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. Feb. 2018, pp. 1–5. DOI: 10.1109/NTMS.2018.8328675.

[96] Alessandro Brighente, Mauro Conti, and Izza Sadaf. "Tell Me How You Re-Charge, I Will Tell You Where You Drove To: Electric Vehicles Profiling Based on Charging-Current Demand". In: *Computer Security - ESORICS 2021 - 26th European Symposium on Research in Computer Security, Proceedings, Part I*. Ed. by Elisa Bertino, Haya Shulman, and Michael Waidner. Vol. 12972. Lecture Notes in Computer Science. Springer, 2021, pp. 651–667. DOI: 10.1007/978-3-030-88418-5_31.

[97] *Improved security for OCPP 1.6-J*. White Paper. Arnhem, Netherlands: Open Charge Alliance, Mar. 2020.

[98] Pol van Aubel, Erik Poll, and Joost Rijneveld. "Non-Repudiation and End-to-End Security for Electric Vehicle Charging". In: *2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. 2019, pp. 1–5. DOI: 10.1109/ISGTEurope.2019.8905444.

[99] Marko van Eekelen, Erik Poll, Engelbert Hubbers, Fabian van den Broek, and Bárbara Vieira. *An end-to-end security design for smart EV charging for Enexis and ElaadNL*. Dec. 2014.

[100] *ISO/IEC 13888-1. Information security – Non-repudiation – Part 1: General Reference*. International Standard 13888-1:2020 Fourth Edition. International Organization for Standardization and International Electrotechnical Commission, Sept. 2020.

[101] *ISO/IEC 13888-3. Information security – Non-repudiation – Part 3: Mechanisms using asymmetric techniques*. International Standard 13888-3:2020 Fourth Edition. International Organization for Standardization and International Electrotechnical Commission, Sept. 2020.

[102] *Open Charge Point Protocol 2.0.1 - Part 2 - Specification*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, 2020.

[103] *Open Charge Point Protocol 2.0.1 - Part 1 - Architecture & Topology*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, 2020.

[104] Simone Orcioni and Massimo Conti. "EV Smart Charging with Advance Reservation Extension to the OCPP Standard". In: *Energies* 13.12 (2020), p. 3263.

[105] *IEC 63119-1:2019. Information exchange for electric vehicle charging roaming service - Part 1: General*. Edition 1.0. Work in Progress. International Electrotechnical Commission (IEC), TC69. June 2019.

[106] Roland Ferwerda, Michel Bayings, Mart Van der Kam, and Rudi Bekkers. "Advancing E-Roaming in Europe: Towards a Single "Language" for the European Charging Infrastructure". In: *World Electric Vehicle Journal* 9.4 (2018). ISSN: 2032-6653. DOI: 10.3390/wevj9040050.

[107] *Open InterCharge Protocol for Emobility Service Providers*. Version 2.3. Hubject GmbH. Oct. 2020.

[108] *Open InterCharge Protocol for Charge Point Operators*. Version 2.3. Hubject GmbH. Oct. 2020.

[109] *Open Clearing House Protocol*. Version 1.4. smartlab. Aug. 2016. URL: http://www.ochp.eu.

[110] *OCHPdirect extension to the Open Clearing House Protocol*. Version 0.2. smartlab. Aug. 2016. URL: http://www.ochp.eu.

[111] *Open Charge Point Interface (OCPI)*. Version 2.2. NKL. Sept. 2019. URL: https://ocpi-protocol.org.

[112] *eMIP Protocol*. v1.0.14. GIREVE. Dec. 2020. URL: https://www.gireve.com/en/download.

[113] Will Arthur and David Challener. *A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security*. Apress, 2015.

[114] *Trusted Platform Module Library - Part 1: Architecture*. Specification Family 2.0 - Rev. 01.38. Trusted Computing Group, Sept. 2016.

[115] *ISO/IEC 11889. Information technology – Trusted platform module library*. International Standard 11889:2015. Geneva, Switzerland: International Organization for Standardization and International Electrotechnical Commission, Dec. 2015.

[116] *TCG Algorithm Registry*. Revision 01.32. Trusted Computing Group. 2020.

[117] *Trusted Platform Module Library - Part 3: Commands*. Specification Family 2.0 - Rev. 01.38. Trusted Computing Group, Sept. 2016.

[118] *Roll-out of public EV charging infrastructure in the EU*. Transport & Environment, Sept. 2018. URL: `www.transportenvironment.org/publications/roll-out-public-ev-charging-infrastructure-eu`.

[119] Dale Hall and Nic Lutsey. *Emerging Best Practices For Electric Vehicle Charging Infrastructure*. Oct. 2017.

[120] Richard Petri, Markus Springer, Daniel Zelle, Ira McDonald, Andreas Fuchs, and Christoph Krauß. “Evaluation of Lightweight TPMs for Automotive Software Updates over the Air”. In: *4th escar USA*. 2016.

[121] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. “Experimental Security Analysis of a Modern Automobile”. In: *2010 IEEE Symposium on Security and Privacy*. May 2010, pp. 447–462. DOI: `10.1109/SP.2010.34`.

[122] P. Fan, B. Sainbayar, and S. Ren. “Operation Analysis of Fast Charging Stations With Energy Demand Control of Electric Vehicles”. In: *IEEE Transactions on Smart Grid* 6.4 (July 2015), pp. 1819–1826. ISSN: 1949-3053. DOI: `10.1109/TSG.2015.2397439`.

[123] Mark Frazer and Philippe Rivard. *Software update method, apparatus and system*. US Patent App. 10/489,777. Mar. 2005.

[124] Yuqing Ren. *Embedded software update methods and systems for digital devices*. US Patent App. 10/986,258. Mar. 2005.

[125] Hsiang-Chueh Shih and Kuan-Liang Kuo. *Method of safe and recoverable firmware update and device using the same*. US Patent App. 12/468,068. Aug. 2010.

[126] Kyle Hyatt. *Chinese carmaker NIO stops traffic for over an hour in Beijing*. Feb. 2019. URL: `https://www.cnet.com/roadshow/news/nio-concept-traffic-stopped-beijing-software-update/`.

[127] H. Mansor, K. Markantonakis, R. N. Akram, and K. Mayes. “Don’t Brick Your Car: Firmware Confidentiality and Rollback for Vehicles”. In: *2015 10th International Conference on Availability, Reliability and Security*. Aug. 2015, pp. 139–148. DOI: `10.1109/ARES.2015.58`.

[128] D. Dolev and A. Yao. “On the security of public key protocols”. In: *IEEE Transactions on Information Theory* 29.2 (Mar. 1983), pp. 198–208. ISSN: 0018-9448. DOI: `10.1109/TIT.1983.1056650`.

[129] Andy Greenberg. *Hackers Remotely Kill A Jeep On The Highway – with Me In It*. `https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/`. July 2015.

[130] *Open Charge Point Protocol 2.0 - Part 2 - Specification*. Open Standard. Arnhem, Netherlands: Open Charge Alliance, 2018.

[131] Hugo Krawczyk. “SIGMA: The ’SIGn-and-MAc’ Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols”. In: *Advances in Cryptology - CRYPTO 2003*. Ed. by Dan Boneh. Berlin, Heidelberg: Springer Berlin Heidelberg, June 2003, pp. 400–425. ISBN: 978-3-540-45146-4.

[132] Infineon. *A safe for sensitive data in the car: Volkswagen relies on TPM from Infineon*. 2019. URL: `www.infineon.com/cms/en/about-infineon/press/market-news/2019/INFATV201901-030.html`.

[133] *A CMC Profile for AIK Certificate Enrollment*. Specification Version 1.0 - Revision 7. Trusted Computing Group, Mar. 2011.

[134] Stefan Thom. *RazorClamMCU*. Feb. 2017. URL: `https://github.com/LordOfDorks/security-1/tree/master/RazorClamMCU`.

[135] V. Sesha Raghav, V. Gowtham, H. S. Jamadagni, and T. V. Prabhakar. “Data throughput maximization for broadband over Power Line”. In: *2014 Sixth International Conference on Communication Systems and Networks (COMSNETS)*. Jan. 2014, pp. 1–6. DOI: 10.1109/COMSNETS.2014.6734898.

[136] J. Anatory, N. Theethayi, R. Thottappillil, M. M. Kissaka, and N. H. Mvungi. “Broadband Power-Line Communications: The Channel Capacity Analysis”. In: *IEEE Transactions on Power Delivery* 23.1 (Jan. 2008), pp. 164–170. ISSN: 0885-8977. DOI: 10.1109/TPWRD.2007.911001.

[137] Daniel Bogdan, Razvan Bogdan, and M Popa. “Delta Flashing of an ECU in the Automotive Industry”. In: May 2016. DOI: 10.1109/SACI.2016.7507429.

[138] D. K. Nilsson and U. E. Larson. “Secure Firmware Updates over the Air in Intelligent Vehicles”. In: *ICC Workshops - 2008 IEEE International Conference on Communications Workshops*. May 2008, pp. 380–384. DOI: 10.1109/ICCW.2008.78.

[139] D. K. Nilsson, U. E. Larson, and E. Jonsson. “Low-cost key management for hierarchical wireless vehicle networks”. In: *2008 IEEE Intelligent Vehicles Symposium*. June 2008, pp. 476–481. DOI: 10.1109/IVS.2008.4621299.

[140] D. K. Nilsson, L. Sun, and T. Nakajima. “A Framework for Self-Verification of Firmware Updates over the Air in Vehicle ECUs”. In: *2008 IEEE Globecom Workshops*. Nov. 2008, pp. 1–5. DOI: 10.1109/GLOCOMW.2008.ECP.56.

[141] S. M. Mahmud, S. Shanker, and I. Hossain. “Secure software upload in an intelligent vehicle via wireless communication links”. In: *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.* June 2005, pp. 588–593. DOI: 10.1109/IVS.2005.1505167.

[142] Derek Lane Lewis. *Over-the-air vehicle systems updating and associate security protocols*. US Patent 9,464,905. Mar. 2016.

[143] Muhammad Sabir Idrees, Hendrik Schweppe, Yves Roudier, Marko Wolf, Dirk Scheuermann, and Olaf Henniger. “Secure Automotive On-Board Protocols: A Case of Over-the-Air Firmware Updates”. In: *Nets4Cars/Nets4Trains*. Mar. 2011.

[144] National Drug Intelligence Center. *Money Laundering in Digital Currencies. No.2008-R0709-003*. Tech. rep. U.S. Department of Justice, 2008.

[145] Australian Transaction Reports and Analysis Centre (AUSTRAC). *Money Laundering in Australia 2011*. Tech. rep. 2011.

[146] P.L. Chatain, A. Zerzan, W. Noor, N. Dannaoui, and L. de Koker. *Protecting Mobile Money against Financial Crimes: Global Policy Challenges and Solutions*. Directions in Development. World Bank Publications, 2011. ISBN: 9780821386705. URL: http://books.google.de/books?id=jvi9BwJr6TkC.

[147] D. Birk, S. Gajek, F. Grobert, and A. Sadeghi. “Phishing Phishers – Observing and Tracing Organized Cybercrime”. In: *Second International Conference on Internet Monitoring and Protection (ICIMP)*. 2007, pp. 3–3. DOI: 10.1109/ICIMP.2007.33.

[148] David Harley and Andrew Lee. *A Pretty Kettle of Phish. Something phishy in your email? What you need to know about phishing fraud. White paper*. http://www.eset.com/us/resources/white-papers/Pretty_Kettle_of_Phish.pdf. July 2007.

[149] M. DeSantis, C. Dougherty, and M. McDowell. *Understanding and Protecting Yourself Against Money Mule Schemes*. https://www.us-cert.gov/security-publications/understanding-and-protecting-yourself-against-money-mule-schemes. June 2012.

[150] R.J. Bolton and D.J. Hand. “Statistical fraud detection: A review”. In: *Statistical Science* (2002), pp. 235–249.

[151] Clifton Phua, Vincent Lee, Kate Smith-Miles, and Ross Gayler. “A comprehensive survey of data mining-based fraud detection research”. In: *Artificial Intelligence Review* (2005).

[152] Agus Sudjianto, Sheela Nair, Ming Yuan, Aijun Zhang, Daniel Kern, and Fernando Cela-Díaz. “Statistical methods for fighting financial crimes”. In: *Technometrics* 52.1 (2010).

[153] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, and J. Christopher Westland. “Data mining for credit card fraud: A comparative study”. In: *Decision Support Systems* 50 (2011).

[154] York Yannikos, Lukas Graner, Martin Steinebach, and Christian Winter. “Data Corpora for Digital Forensics Education and Research”. In: *Advances in Digital Forensics X - 10th IFIP WG 11.9 International Conference, Revised Selected Papers*. 2014, pp. 309–325. DOI: 10.1007/978-3-662-44952-3_21.

[155] William Jack, Suri Tavneet, and Robert Townsend. “Monetary Theory and Electronic Money: Reflections on the Kenyan Experience”. In: *Economic Quarterly* 96 (2010). First Quarter 2010.

[156] Y. Sahin and E. Duman. “Detecting Credit Card Fraud by Decision Trees and Support Vector Machines”. In: *International MultiConference of Engineers and Computer Scientists*. 2011.

[157] Emilie Lundin, Hakan Kvarnström, and Erland Jonsson. “A Synthetic Fraud Data Generation Methodology”. In: *Information and Communications Security*. Ed. by Robert Deng, Feng Bao, Jianying Zhou, and Sihan Qing. Vol. 2513. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, pp. 265–277. ISBN: 978-3-540-00164-5. DOI: 10.1007/3-540-36159-6_23.

[158] E.L. Barse, H. Kvarnström, and E. Jonsson. “Synthesizing test data for fraud detection systems”. In: *19th Annual Computer Security Applications Conference*. 2003, pp. 384–394.

[159] E.A. Lopez-Rojas and S. Axelsson. “Multi Agent Based Simulation (MABS) of Financial Transactions for Anti Money Laundering (AML)”. In: *Nordic Conference on Secure IT Systems*. Blekinge Institute of Technology. 2012.

[160] Chrystel Gaber. “Sécurisation d’un système de transactions sur terminaux mobiles”. PhD thesis. Université de Caen Basse-Normandie, Oct. 2013.

[161] Raman Kazhamiakin, Marco Pistore, and Luca Santuari. “Analysis of communication models in web service compositions”. In: *World Wide Web (WWW 2006)*. Edinburgh, Scotland: ACM, 2006, pp. 267–276. ISBN: 1-59593-323-9.

[162] Thierry Massart and Cédric Meuter. *Efficient online monitoring of LTL properties for asynchronous distributed systems*. Tech. rep. Université Libre de Bruxelles, 2006.

[163] N. Delgado, A.Q. Gates, and S. Roach. “A taxonomy and catalog of runtime software-fault monitoring tools”. In: *IEEE Transactions on Software Engineering* 30.12 (2004), pp. 859–872. ISSN: 0098-5589. DOI: 10.1109/TSE.2004.91.

[164] Josef Schiefer, Szabolcs Rozsnyai, Christian Rauscher, and Gerd Saurer. “Event-Driven Rules for Sensing and Responding to Business Situations”. In: *DEBS*. 2007, pp. 198–205. URL: http://doi.acm.org/10.1145/1266894.1266934.

[165] S. Tjoa, S. Jakoubi, G. Goluch, G. Kitzler, S. Goluch, and G. Quirchmayr. “A Formal Approach Enabling Risk-aware Business Process Modeling and Simulation”. In: *IEEE Transactions on Services Computing* 4.2 (2011), pp. 153–166.

[166] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. “From Data Mining to Knowledge Discovery in Databases”. In: *AI Magazine* 17.3 (1996), pp. 37–54.

[167] Wil M. P. van der Aalst. “Business Process Management: A Comprehensive Survey”. In: *ISRN Software Engineering* (2013), p. 37.

[168] Fabrizio Maria Maggi, Marco Montali, Michael Westergaard, and Wil M. P. van der Aalst. “Monitoring Business Constraints with Linear Temporal Logic: An Approach Based on Colored Automata”. In: *Business Process Management (BPM 2011)*. Vol. 6896. LNCS. Springer, 2011, pp. 132–147.

[169] P. Ochsenschläger, J. Repp, R. Rieke, and U. Nitsche. “The SH-Verification Tool – Abstraction-Based Verification of Co-operating Systems”. In: *Formal Aspects of Computing, The International Journal of Formal Method* 10 (4 1998), pp. 381–404. ISSN: 0934-5043. DOI: 10.1007/s001650050023.

[170] M. Aston, S. McCombie, B. Reardon, and P. Watters. “A Preliminary Profiling of Internet Money Mules: An Australian Perspective”. In: *Ubiquitous, Autonomic and Trusted Computing, 2009. UIC-ATC’09. Symposia and Workshops on*. July 2009, pp. 482–487. DOI: 10.1109/UIC-ATC.2009.63.

[171] Marc Llanes, Elsa Prieto, Rodrigo Diaz, Luigi Coppolino, Antonio Sergio, Rosario Cristaldi, Mohammed Achemlal, Said Gharout, Chrystel Gaber, Andrew Hutchison, and Keiran Dennie. *Scenario Requirements (Public version)*. Tech. rep. FP7-257475 MASSIF European project, Apr. 2011.

[172] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. “The WEKA data mining software: an update”. In: *ACM SIGKDD Explorations Newsletter* 11.1 (2009), pp. 10–18.

[173] Eibe Frank and Ian H Witten. "Generating accurate rule sets without global optimization". In: (1998).

[174] John Ross Quinlan. *C4.5: programs for machine learning*. Vol. 1. Morgan Kaufmann, 1993.

[175] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[176] Joseck Luminzu Mudiri. *Fraud in Mobile Financial Services*. Tech. rep. MicroSave, 2012.

[177] B. Walzel, M. Hirz, H. Brunner, and N. Kreutzer. "Robot-Based Fast Charging of Electric Vehicles". In: *WCX SAE World Congress Experience*. Apr. 2019. DOI: 10.4271/2019-01-0869.

[178] *SHE – Secure Hardware Extension Functional Specification*. Specification. Hersteller Initiative Software (HIS) Consortium, 2009.

[179] Marko Wolf and Timo Gendrullis. "Design, Implementation, and Evaluation of a Vehicular Hardware Security Module". In: *Information Security and Cryptology (ICISC 2011)*. Ed. by Howon Kim. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 302–318. ISBN: 978-3-642-31912-9.

[180] André Groll, Jan Holle, Marko Wolf, and Thomas Wollinger. "Next generation of automotive security: secure hardware and secure open platforms". In: (Jan. 2010).

[181] *Protection Profile V2X Hardware Security Module, Release 1.4.0*. Tech. rep. CAR 2 CAR Communication Consortium (C2C CC), 2019.

[182] *Trusted Platform Module Library Specification*. Family 2.0, Level 00, Revision 01.38. Trusted Computing Group. Sept. 2016.

[183] *ISO/IEC preFDIS 15118-20. Road vehicles – Vehicle-to-Grid Communication Interface – Part 20: 2nd generation network and application protocol requirements*. preFDIS 15118-20. International Organization for Standardization and International Electrotechnical Commission, Jan. 2019.

[184] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC 5280. RFC Editor, May 2008.

[185] J. Petit, M. Feiri, and F. Kargl. "Revisiting attacker model for smart vehicles". In: *IEEE 6th International Symposium on Wireless Vehicular Communications*. Sept. 2014, pp. 1–5. DOI: 10.1109/WIVEC.2014.6953258.

[186] *EV Charging Systems – Security Threats*. Tech. rep. European Network for Cyber Security, Apr. 2016.

[187] *ISO/IEC 29115:2013. Information technology – Security techniques – Entity authentication assurance framework*. International Standard. Edition 1. International Organization for Standardization and International Electrotechnical Commission, Apr. 2013.

[188] E. Barker. *Special Publication 800-57 Part 1, Recommendation for Key Management: General*. Tech. rep. National Institute of Standards and Technology NIST, 2016.

[189] *TCG TPM v2.0 Provisioning Guidance*. Guidance Ver. 1.0 - Rev. 1.0. Trusted Computing Group, Mar. 2017.

[190] D. Moghimi, B. Sunar, T. Eisenbarth, and N. Heninger. "TPM-FAIL: TPM meets Timing and Lattice Attacks". In: *29th USENIX Security Symposium (USENIX Security 20)*. Boston, MA: USENIX Association, Aug. 2020.

[191] Infineon. *OPTIGATM TPM SLI 9670*. https://www.infineon.com/cms/en/product/security-smart-card-solutions/optiga-embedded-security-solutions/optiga-tpm/sli-9670/. Feb. 2019.

[192] *Open InterCharge Protocol for Emobility Service Provider / Charge Point Operator*. Version 2.2. Hubject GmbH. Oct. 2017.

[193] M. Nystrom and B. Kaliski. *PKCS #10: Certification Request Syntax Specification Version 1.7*. RFC 2986. RFC Editor, Nov. 2000.

[194] B. Berman. *The ISO standard for electric-vehicle "Plug-and-Charge" faces security concerns*. Ed. by SAE International. Aug. 2020. URL: https://www.sae.org/news/2020/08/iso-ev-plug-and-charge-standard-faces-security-concerns.

[195] *Hubject Plug&Charge Certificate Policy (CP) for the Hubject ISO 15118 V2G PKI*. Tech. rep. Version 1.7. Hubject GmbH, June 2020.

[196] L. Popa, B. Groza, and P. Murvay. "Performance evaluation of elliptic curve libraries on automotive-grade microcontrollers". In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 2019, pp. 1–7.

[197] *Specification of Secure Onboard Communication*. Tech. rep. AUTOSAR, 2017.

[198] T. Zhao, C. Zhang, L. Wei, and Y. Zhang. “A secure and privacy-preserving payment system for Electric vehicles”. In: *Communications (ICC), 2015 IEEE International Conference on*. IEEE. 2015, pp. 7280–7285.

[199] Cheng Xu, Hongzhe Liu, Peifeng Li, and Pengfei Wang. “A Remote Attestation Security Model Based on Privacy-Preserving Blockchain for V2X”. In: *IEEE Access* 6 (2018), pp. 67809–67818.

[200] Dipayan P Ghosh, Robert J Thomas, and Stephen B Wicker. “A privacy-aware design for the vehicle-to-grid framework”. In: *2013 46th Hawaii International Conference on System Sciences*. IEEE. 2013, pp. 2283–2291.

[201] Neetesh Saxena, Santiago Grijalva, Victor Chukwuka, and Athanasios V Vasilakos. “Network security and privacy challenges in smart vehicle-to-grid”. In: *IEEE Wireless Communications* 24.4 (2017), pp. 88–98.

[202] Shuai Wang, Baoyi Wang, and Shaomin Zhang. “A Secure Solution of V2G Communication Based on Trusted Computing”. In: *12th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID)*. IEEE. 2019, pp. 98–102.

[203] *TCG Mobile Trusted Module Specification*. Specification Version 1.0 - Rev. 7.02. Trusted Computing Group, Apr. 2010.

[204] Jianxiong Shao, Yu Qin, Dengguo Feng, and Weijin Wang. “Formal Analysis of Enhanced Authorization in the TPM 2.0”. In: *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*. ASIA CCS’15. Singapore, Republic of Singapore: ACM, 2015, pp. 273–284. ISBN: 978-1-4503-3245-3. DOI: `10.1145/2714576.2714610`.

[205] *ISO/IEC Directives, Part 1. Consolidated ISO Supplement – Procedures for the technical work – Procedures specific to ISO*. Tech. rep. Geneva, Switzerland: International Organization for Standardization, 2021.

[206] SAE International Surface Vehicle Recommended Practice. *Cybersecurity Guidebook for Cyber-Physical Vehicle Systems*. SAE Standard J3061. Society of Automotive Engineers (SAE). Jan. 2016.

[207] *ISO 26262. Road vehicles – Functional safety*. International Standard. Geneva, Switzerland: International Organization for Standardization, 2018.

[208] *ISO/SAE 21434. Road vehicles – Cybersecurity engineering*. International Standard. International Organization for Standardization and SAE International, 2021.

[209] Georg Macher, Richard Messnarz, Eric Armengaud, Andreas Riel, Eugen Brenner, and Christian Kreiner. “Integrated Safety and Security Development in the Automotive Domain”. In: Mar. 2017. DOI: `10.4271/2017-01-1661`.

[210] Georg Macher, Harald Sporer, Eugen Brenner, and Christian Kreiner. “An Automotive Signal-Layer Security and Trust-Boundary Identification Approach”. In: *Procedia Computer Science* 109 (2017). 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal, pp. 490–497. ISSN: 1877-0509. DOI: `https://doi.org/10.1016/j.procs.2017.05.317`.

[211] Andreas Riel, Christian Kreiner, Georg Macher, and Richard Messnarz. “Integrated design for tackling safety and security challenges of smart products and digital manufacturing”. In: *CIRP Annals - Manufacturing Technology* 66 (2017), pp. 177–180. URL: `ff10.1016/j.cirp.2017.04.037ff`.

[212] Andrew F. Blum and R. Thomas Jr. Long. *Hazard Assessment of Lithium Ion Battery Energy Storage Systems*. Final Report. Fire Protection Research Foundation, Feb. 2016.

[213] F. Sagstetter, M. Lukasiewycz, S. Steinhorst, M. Wolf, A. Bouard, W. R. Harris, S. Jha, T. Peyrin, A. Poschmann, and S. Chakraborty. “Security challenges in automotive hardware/software architecture design”. In: *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*. Mar. 2013, pp. 458–463.

[214] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner. “SAHARA: A security-aware hazard and risk analysis method”. In: *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*. Mar. 2015, pp. 621–624.

[215] Sophie Peresson. “Counterfeit automotive parts increasingly putting consumer safety at risk”. In: *WTR* (2019). URL: `www.worldtrademarkreview.com/anti-counterfeiting/counterfeit-automotive-parts-increasingly-putting-consumer-safety-risk`.

[216] *Hardware Requirements for a Device Identifier Composition Engine*. Specification Family 2.0 - Level 00 Revision 78. Trusted Computing Group, Mar. 2018.

[217] M. Brandl, H. Gall, M. Wenger, V. Lorentz, M. Giegerich, F. Baronti, G. Fantechi, L. Fanucci, R. Roncella, R. Saletti, S. Saponara, A. Thaler, M. Cifrain, and W. Prochazka. "Batteries and battery management systems for electric vehicles". In: *Design, Automation Test in Europe Conference Exhibition (DATE)*. 2012.

[218] Levi Tillemann and Colin McCormick. "The Faster, Cheaper, Better Way to Charge Electric Vehicles". In: *WIRED* (2018). URL: `www.wired.com/story/the-faster-cheaper-better-way-to-charge-electric-vehicles/`.

[219] Georg Macher, Andrea Höller, Harald Sporer, Eric Armengaud, and Christian Kreiner. "A Combined Safety-Hazards and Security-Threat Analysis Method for Automotive Systems". In: *Computer Safety, Reliability, and Security*. 2015.

[220] Bryce Gaton. "NZ company offers solution to Nissan Leaf owners wanting a bigger battery". In: *The Driven* (2019). URL: `thedriven.io/2019/08/13/nz-company-offers-solution-to-nissan-leaf-owners-wanting-a-bigger-battery/`.

[221] Lukas Jäger, Richard Petri, and Andreas Fuchs. "Rolling DICE: Lightweight Remote Attestation for COTS IoT Hardware". In: *Proceedings of the 12th International Conference on Availability, Reliability and Security*. ARES. Reggio Calabria, Italy, 2017. ISBN: 9781450352574.

[222] Hannes Tschofenig and Manuel Pégourié-Gonnard. "Performance investigations". In: *IETF proceeding92* (2015).

[223] Chris M. Lonvick and Tatu Ylonen. *The Secure Shell (SSH) Transport Layer Protocol*. RFC 4253. Jan. 2006. URL: `https://rfc-editor.org/rfc/rfc4253.txt`.

[224] *PC Client Platform TPM Profile (PTP) Specification*. Family 2.0, Revision 00.43. Trusted Computing Group. Jan. 2015.

[225] Ishtiaq Rouf, Rob Miller, Hossen Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe, and Ivan Seskar. "Security and Privacy Vulnerabilities of In-car Wireless Networks: A Tire Pressure Monitoring System Case Study". In: *Proceedings of the 19th USENIX Conference on Security*. USENIX Security'10. Washington, DC: USENIX Association, 2010, pp. 21–21.

[226] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. "Comprehensive Experimental Analyses of Automotive Attack Surfaces". In: *Proceedings of the 20th USENIX Conference on Security*. SEC'11. San Francisco, CA: USENIX Association, 2011, pp. 6–6.

[227] Charlie Miller and Chris Valasek. *Adventures in Automotive Networks and Control Units*. 2014.

[228] Gianpiero Costantino and Ilaria Matteucci. *KOFFEE - Kia OFFensivE Exploit*. Tech. rep. National Research Council CNR, 2020.

[229] Charlie Miller and Chris Valasek. *Remote Exploitation of an Unaltered Passenger Vehicle*. Tech. rep. Illmatics, 2015.

[230] Sen Nie, Ling Liu, and Yuefeng Du. *Free-Fall: Hacking Tesla From Wireless To CAN Bus*. Tech. rep. Keen Security Lab of Tencent, 2017.

[231] CENELEC - European Committee for Electrotechnical Standardization. *EN50126 - Railway applications – The specification and demonstration of Reliability, Availability, Maintainablity and Safety (RAMS) Part 1: Basic requirements and generic process*. EN 50126:1999 E. CENELEC Central Secretariat, 2010.

[232] *Elektrische Bahn-Signalanlagen, Teil 200: Sicheres Übertragungsprotokoll RaSTA nach DIN EN 50159 (VDE 0831-159)*. Tech. rep. DIN VDE V 0831-200:2015-06. VDE Verband der Elektrotechnik Elektronik Informationstechnik e. V. (VDE), 2015.

[233] Marco Rocchetto and Nils Ole Tippenhauer. "On Attacker Models and Profiles for Cyber-Physical Systems". In: *Computer Security – ESORICS 2016: 21st European Symposium on Research in Computer Security, Heraklion, Greece, September 26-30, 2016, Proceedings, Part II*. Ed. by Ioannis Askoxylakis, Sotiris Ioannidis, Sokratis Katsikas, and Catherine Meadows. Springer International Publishing, 2016, pp. 427–449. ISBN: 978-3-319-45741-3. DOI: `10.1007/978-3-319-45741-3_22`.

[234] Marcus K. Rogers. "A Two-dimensional Circumplex Approach to the Development of a Hacker Taxonomy". In: *Digital Investigation* 3.2 (June 2006), pp. 97–102. ISSN: 1742-2876. DOI: `10.1016/j.diin.2006.03.001`.

[235] C. A. Meyers, S. S. Powers, and D. M. Faissol. *Taxonomies of Cyber Adversaries and Attacks: A Survey of Incidents and Approaches*. Oct. 2009. DOI: 10.2172/967712.

[236] P. von Oppenkowski. *Analyse sicherheitsrelevanter Geschäftsprozesse eines Anwendungsfalls aus der Finanzbranche und Ermittlung der hierfür geeigneten Methoden*. 2011. ISBN: 9783842822931. URL: https://books.google.de/books?id=jXVhAQAAQBAJ.

[237] Ryan Seebruck. "A Typology of Hackers". In: *Digital Investigation* 14.C (Sept. 2015), pp. 36–45. ISSN: 1742-2876. DOI: 10.1016/j.diin.2015.07.002.

[238] Raoul Chiesa. *Hackers Profiling: Who are the Attackers?* URL: http://f3magazine.unicri.it/?p=306.

[239] Ronald S. Ross. *Special Publication 800-30. Guide for Conducting Risk Assessments*. Tech. rep. Revision 1. Gaithersburg, MD, United States: National Institute of Standards & Technology (NIST), 2012.

[240] *Critical Infrastructure Protection. Sector-Specific Agencies Need to Better Measure Cybersecurity Progress*. Tech. rep. GAO-16-79. United States Government Accountability Office, 2015. URL: http://www.gao.gov/products/GAO-16-79.

[241] Jack A. Jones. *An Introduction to Factor Analysis of Information Risk (FAIR). A framework for understanding, analyzing, and measuring information risk*. Tech. rep. Draft. 2005.

[242] Kathleen Ann Uradnik, Lori A. Johnson, and Sara Hower. *Battleground: Government and Politics*. Vol. 1. Battleground Europe. Greenwood, 2011. ISBN: 9780313343131. URL: https://books.google.de/books?id=uarFTBpg11wC.

[243] J. Matusitz. *Terrorism and Communication: A Critical Introduction*. SAGE Publications, 2012. ISBN: 9781452289557. URL: https://books.google.de/books?id=iY0gAQAAQBAJ.

[244] Dorothy E. Denning. *Cyberterrorism*. http://www.stealth-iss.com/documents/pdf/CYBERTERRORISM.pdf. 2000.

[245] K. Jack Riley. *Terrorism and Rail Security*. Tech. rep. CT-224. RAND Corporation Testimony, 2004. URL: https://www.rand.org/pubs/testimonies/CT224.html.

[246] *Performance Audit on Security Management in Indian Railways*. Tech. rep. Tech. Rep. 14 of 2011-12 (Railways). 2011. URL: cag.gov.in/content/report-no-14-2011-performance-audit-security-management-indian-railways-union-government.

[247] *Advanced Persistent Threats: A Symantec Perspective*. 2011. URL: http://www.symantec.com/content/en/us/enterprise/white_papers/b-advanced_persistent_threats_WP_21215957.en-us.pdf.

[248] *W32.Stuxnet Dossier*. 2011. URL: http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf.

[249] Olivier Thonnard, Leyla Bilge, Anand Kashyap, and Martin Lee. "Are You at Risk? Profiling Organizations and Individuals Subject to Targeted Attacks". In: *19th International Conference on Financial Cryptography and Data Security (FC 2015), Revised Selected Papers*. 2015, pp. 13–31. DOI: 10.1007/978-3-662-47854-7_2.

[250] Salvatore J. Stolfo, Steven M. Bellovin, Angelos D. Keromytis, Shlomo Hershkop, Sean W. Smith, and Sara Sinclair. *Insider Attack and Cyber Security: Beyond the Hacker*. Advances in Information Security. Springer US, 2008. ISBN: 978-0-387-77322-3. URL: https://books.google.de/books?id=tW298SIeg4IC.

[251] *HMG IA Standard No. 1. Technical Risk Assessment*. Tech. rep. Issue No. 3.51. 2009.

[252] *Securing Control and Communications Systems in Rail Transit Environments*. Tech. Rep. APTA-SS-CCS-RP-002-13. American Public Transportation Association, 2013. URL: http://www.apta.com/resources/standards/documents/apta-ss-ccs-rp-002-13.pdf.

[253] Jim Henderson. *Insider Threat Incidents. Could They Happen To Your Organization?* Mar. 2017. URL: www.nationalinsiderthreatsig.org/pdfs/Insider%5C%20Threats%5C%20Incidents-Could%5C%20They%5C%20Happen%5C%20To%5C%20Your%5C%20Organization.pdf.

[254] M.E. Kabay. *Attacks on power systems: Data leakage, espionage, insider threats, sabotage*. http://www.networkworld.com/article/2217680/data-center/attacks-on-power-systems--data-leakage--espionage--insider-threats--sabotage.html. Sept. 2010.

[255] Chris Cain. "Controlling Vendor Access For Small Business". In: (Sept. 2013).

[256] CENELEC - European Committee for Electrotechnical Standardization. *EN50128 - Railway applications – Communications, signalling and processing systems – Software for railway control and protection systems*. EN 50128:2001 E. CENELEC Central Secretariat, 2010.

[257] CENELEC - European Committee for Electrotechnical Standardization. *EN50129 - Railway applications – Communications, signalling and processing systems – Safety related electronic systems for signalling*. EN 50129:2003. CENELEC Central Secretariat, 2003.

[258] *DIN VDE V 0831-200. Electric signalling systems for railways - Part 200: Safe transmission protocol according to DIN EN 50159*. Tech. rep. DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE, 2015.

[259] *Securing Control and Communications Systems in Rail Transit Environments – Part II: Defining a Security Zone Architecture for Rail Transit and Protecting Critical Zones*. Tech. rep. APTA-SS-CCS-RP-002-13. APTA Control and Communications Working Group, 2013.

[260] Hans-Hermann Bock, Jens Braband, Birgit Milius, and Hendrik Schäbe. "Towards an IT security protection profile for safety-related communication in railway automation". In: *Computer Safety, Reliability, and Security*. Springer, 2012, pp. 137–148. DOI: `10.1007/978-3-642-33678-2_12`.

[261] Suvda Myagmar, Adam J Lee, and William Yurcik. "Threat modeling as a basis for security requirements". In: *Symposium on requirements engineering for information security (SREIS)*. Vol. 2005. 2005, pp. 1–8.

[262] N. Asokan, J. E. Ekberg, K. Kostiainen, A. Rajan, C. Rozas, A. R. Sadeghi, S. Schulz, and C. Wachsmann. "Mobile Trusted Computing". In: *Proceedings of the IEEE* 102.8 (Aug. 2014), pp. 1189–1206. ISSN: 0018-9219. DOI: `10.1109/JPROC.2014.2332007`.

[263] Andreas Fuchs, Christoph Krauß, and Jürgen Repp. "Advanced Remote Firmware Upgrades Using TPM 2.0". In: *Proceedings of the 31th International Conference on ICT Systems Security and Privacy Protection (IFIP SEC)*. 2016.

[264] Sergey Tverdyshev, Holger Blasum, Bruno Langenstein, Jonas Maebe, Bjorn De Sutter, Bertrand Leconte, Benoît Triquet, Kevin Müller, Michael Paulitsch, Axel Söding-Freiherr von Blomberg, and Axel Tillequin. *MILS Architecture*. EURO-MILS, 2013. DOI: `10.5281/zenodo.45164`.

[265] *TSS System Level API and TPM Command Transmission Interface Specification*. Trusted Computing Group. 2016.

[266] *TSS TAB and Resource Manager*. Trusted Computing Group. 2016.

[267] Andreas Fuchs, Henk Birkholz, Ira McDonald, and Carsten Bormann. *Time-Based Uni-Directional Attestation*. Internet-Draft. The Internet Engineering Task Force (IETF), Oct. 2017. URL: `https://datatracker.ietf.org/doc/draft-birkholz-i2nsf-tuda/`.

[268] Rodrigo Vieira Steiner and Emil Lupu. "Attestation in Wireless Sensor Networks: A Survey". In: *ACM Computing Surveys* 49.3 (Sept. 2016), 51:1–51:31. ISSN: 0360-0300. DOI: `10.1145/2988546`.

[269] A. M. Azab, P. Ning, E. C. Sezer, and X. Zhang. "HIMA: A Hypervisor-Based Integrity Measurement Agent". In: *Annual Computer Security Applications Conference (ACSAC'09)*. Dec. 2009, pp. 461–470. DOI: `10.1109/ACSAC.2009.50`.

[270] Bryan D. Payne, Martim Carbone, and Wenke Lee. "Secure and flexible monitoring of virtual machines". In: *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC 2007)*. 2007. DOI: `10.1109/acsac.2007.10`.

[271] D. Garlan, S. W. Cheng, A. C. Huang, B. Schmerl, and P. Steenkiste. "Rainbow: Architecture-based Self-adaptation with Reusable Infrastructure". In: *Computer* 37.10 (Oct. 2004), pp. 46–54. ISSN: 0018-9162. DOI: `10.1109/MC.2004.175`.

[272] M. Dinkel, S. Stesny, and U. Baumgarten. "Interactive Self-Healing for Black-Box Components in Distributed Embedded Environments". In: *Communication in Distributed Systems (KiVS), 2007 ITG-GI Conference*. Feb. 2007, pp. 1–12.

[273] Daniel Mellado, Eduardo Fernández-Medina, and Mario Piattini. "Applying a Security Requirements Engineering Process". In: *Computer Security – ESORICS 2006*. Ed. by Dieter Gollmann, Jan Meier, and Andrei Sabelfeld. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 192–206. ISBN: 978-3-540-44605-7.

[274] Ambrosio Toval, Joaquín Nicolás, Begoña Moros, and O García. "Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach". In: *Requirements Engineering Journal* 6 (2001), pp. 205–219.