2018-01-0012 Published 03 Apr 2018

Leveraging Hardware Security to Secure Connected Vehicles

Christopher Corbett AUDI AG

Martin Brunner Infineon Technologies AG

Karsten Schmidt and Rolf Schneider AUDI AG

Udo Dannebaum Infineon Technologies AG

Citation: Corbett, C., Brunner, M., Schmidt, K., Schneider, R. et al., "Leveraging Hardware Security to Secure Connected Vehicles," SAE Technical Paper 2018-01-0012, 2018, doi:10.4271/2018-01-0012.

Abstract

Advanced safety features and new services in connected cars depend on the security of the underlying vehicle functions. Due to the interconnection with the outside world and as a result of being an embedded system a modern vehicle is exposed to both, malicious activities as faced by traditional IT world systems as well as physical attacks. This introduces the need for utilizing hardware-assisted security measures to prevent both kinds of attacks.

In this paper we present a survey of the different classes of hardware security devices and depict their different functional range and application. We demonstrate the feasibility of our approach by conducting a case study on an exemplary implementation of a function-on-demand use case. In particular, our example outlines how to apply the different hardware security approaches in practice to address real-world security topics.

We conclude with an assessment of today's hardware security devices. Based on our presented case study we outline the identified gaps and derive the necessary future developments for next-generation hardware security devices to meet the requirements for automotive applications.

Introduction

The automotive industry is currently driven by three trends that radically change the traditional business model:

1. CO2-neutral mobility requires alternative drive trains. This includes the shift from combustion engines powered by fossil fuels towards emission-free driving utilizing alternative drive technologies, such as electric powertrains and efficient engines powered by synthetic liquid fuels from renewable energy sources.
2. Autonomous driving will change the technical- as well as the design-perspective of vehicles by enabling new use cases to the driver and opening up new business models for the OEM.
3. Innovative mobility concepts will drastically change the role and usage of vehicles, which become shared assets being seemingly integrated into people's daily life and thus need to be constantly interconnected and "always on".

Cross-vehicle communication and the continuous integration of the car with its environment enable new services for OEMs. To utilize these opportunities, OEMs need to establish trusted mobility-platforms and furthermore not only build and sell vehicles, but also act as a mobility provider.

Outstanding examples include online services for connected vehicles, V2X communication and advanced driving assistance systems. Environmental data, such as traffic prediction and geographic information, help to improve driving strategies and thereby enables an overall reduction in fuel consumption. Furthermore wireless access to the vehicle enables remote services such as diagnostics and software updates instead of costly recalls.

With the increasing number of functions more ECUs are added to vehicles. As a result, today's high-end passenger vehicles have up to 130 ECUs. Future automotive systems are expected to intensify the trend of interconnection, since multiple functions are merged to a few number of high performance domain controller ECUs. Hence, additional aspects have to be taken into account:

- Converge to a high performance domain controller ECU backbone (e.g. AUDI zFAS)
- High level of integration and new software architecture concepts
- Standardization of protocols for network and interfaces.
- High-speed communication (payload and latency)

- Software will be driver for innovation and serve as a unique selling point
- Software with user interaction needs to be upgradable in order to add new functions or to fix bugs.
- Separation of basic driving functions and functions with user interaction

The remainder of this paper is structured as follows: In section 2 we discuss the need for trust anchors to protect connected vehicles. We outline the need for hardware based security and discuss the manifold aspects of implementing trust anchors in hardware. In section 3 we give a survey on hardware security devices for automotive applications. In particular we describe several available modules and their application. In section 4 we present a case study for a "function on demand" use case. Our findings along with an outlook on future requirements conclude the paper.

Need for Trust Anchors to Protect Connected Vehicles

There is an increasing demand for interconnection and digitalization resulting from the three megatrends as innovation drivers - as outlined in the introduction. This in turn creates requirements for higher bandwidth, processing of larger amounts of data and increasing flexibility in future vehicle architectures.

From a security perspective the increase in attack surface and code size imply the following two axioms:

1. Anything that is connected can (and will) be attacked
2. Complexity is the enemy of security

Axiom 1: Anything That Is Connected Can (and Will) Be Attacked

New vehicle architectures for service-oriented communication have made a big step towards implementation and mass usage for both in-vehicle connections and connections to the outside world. The connection to a car manufacturer's cloud creates data transfer in both directions. Thus, the formerly closed automotive systems become now part of an open system. Due to their interconnection with other - potentially untrusted -systems outside the car (e.g. on the internet) they are exposed to malicious activities as faced by traditional IT world systems.

In addition, security and safety are intrinsically linked, since a security incident may also impact safety mechanisms: While safety comprises modeling of failures based on the availability of physical models (and physics do not change), security involves thinking like an attacker. Furthermore no reliable models are available due to humans being involved as part of the technical systems. The unpredictable human factor, continuous technical evolution along with a constantly changing threat landscape makes security a challenging topic. The lack of common industry standards and the missing „security business case" (i.e. security is neither a customer perceivable function nor a marketable feature) aggravates this situation.

Consequently security can be seen as a prerequisite for safety despite having identical consequences (e.g. harm to life). As experiences from other industries show, whatever is accessible will be (successfully) attacked sooner or later [1], [2], [3], [4], [5]. Yet, there is no large-scale business case for professional attackers in the automotive domain. However, the more interconnected vehicles get and the more sensitive data are stored on them, the higher the potential profit will be. As a result the attractiveness for potential attackers will increase. Motives could be e.g.

- simple monetary reasons (data abuse, identity theft, ransomware)
- extortion attempts which threaten reputation loss (for example all vehicles of a given brand worldwide stop driving and start blinking and honking at the same time)
- industry espionage by competitors
- terrorist usage by utilizing cars as weapons

In recent years the security of connected vehicles attracted the interest of the security community. As a result, vehicle attacks were on the rise (e.g. as documented in [6]).

An exemplary listing of attacks against vehicles, that recently gained public attention includes:

- Attack due to a lack of encryption in the cellular connection. Instead of using secure HTTPS, data were exchanged via the unencrypted HTTP protocol. More than 2 million cars have been affected. The error could get fixed by a remote over the air software update.
- A total number of 1.4 million vehicles were voluntarily recalled to prevent them from being remotely compromised. An open port in the Head Unit allowed access to the E/E network and thereby opened the vehicle (audio, HVAC, engine cut-off, steering at low-speed) [7] for being controlled remotely
- Attack that involved creating a man in the middle attack between smartphone - infotainment - back end and a spoof of the smartphone application issuing commands to the car, such as remotely unlocking the doors. As a result the app had been patched and users had to download a new version via the app store.
- Six significant bugs could allow attackers to take control of vehicles resulting in safety implications for drivers. The attackers had to physically access the vehicle first, which made it more difficult than many other hacks. Once they were connected through an Ethernet cable, they were later able to access the systems from afar. This allowed them to take control of the screens. They were able to manipulate the speedometer to show the wrong speed, lower and raise windows, lock and unlock the car

and turn the car on or off. The error could get fixed by a remote over the air software update.

All these attacks share the fact that they were all primarily software-based and they outlined that an initially local attack may lead to remote attacks (e.g. though physical access is needed to perform the initial attack, it may be reproduced remotely or remote access may be established later on).

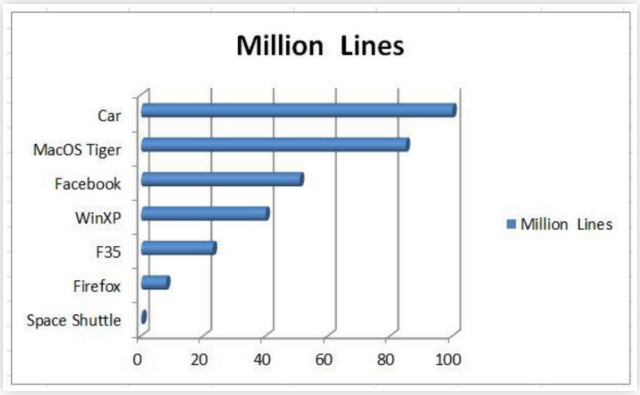
For the car as an embedded system, it must be assumed that a potential attacker may have full physical access for an unlimited amount of time.

This introduces new security challenges for the design of automotive vehicle networks since it also (indirectly) impacts already implemented safety mechanisms, which are usually still based on a closed world view regarding the vehicle network.

Axiom 2: Complexity Is the Enemy of Security

Based on the expanding interconnection in vehicle networks the implementation of new services and applications further increases the complexity of the code base, containing over 100 million lines of code in a modern vehicle [8]. Also the underlying value chain, code coming from a variety of different suppliers and the fact that a modern high-end vehicle may have more than 130 ECUs (from several suppliers with individual cross-functions and stakeholders) have a huge impact on the overall complexity. The source code quality serves as an indicator for the level of security of certain software. Assuming only one error per 4000 lines of code, this would mean that there are approximately 25000 possible vulnerabilities per vehicle. While this can't be extrapolated one-to-one since vulnerability research is way more complex, it still outlines the potential attack surface. Figure 1 sets this number in relation to the code size of some examples of well-known software products and systems. For instance, the primary flight software of the NASA space shuttle had ~ 400,000 lines of code, the Apple operating system Tiger has 80 million lines of code, and Microsoft Windows XP has 40 million lines of code. Another example from fauna: The entire genome of a mouse has 120 million base pairs [9].

FIGURE 1 Comparison of code sizes (in million lines of code)



Besides the known challenges for deploying security measures in the automotive domain (e.g. the long life-cycle and long-term sustainability of automotive security architectures, continuously changing requirements and threats, costs, personalization of devices and keys along the supply chain, etc.) the primary issue becomes complexity. Complexity introduces an increased error-proneness and this in return increases the number of possible vulnerabilities.

The Need for Hardware-Based Security

A raising number of innovations within automotive systems are achieved by software. The hardware serves thereby as a sustainable platform for creating new use and business cases. This raises several issues which are described in the following.

A vehicle has a comparably long life time. Thus, its security architecture has to be sustained up to 24 years (Figure 2), beginning from the time the concept was developed until the car is put out of operation.

It is expected that during this life time new use cases must be supported even if they were not taken into account at the design time of the security architecture. Furthermore the threat landscape will change constantly and (likely) rapidly during this time frame. This raises the demand for spending constant efforts to sustain the security level for this long period of time.

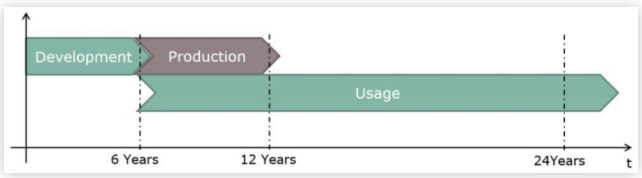
Vehicle users expect the same seamless interoperability, flexibility and ease-of-use as known from all other (connected) devices they are used to deal with on consumer level (e.g. Smartphone, Tablet, etc.) as well as a user of "cyber-physical systems", such as smart-home systems.

At the same time the user expects to be secured and privacy-protected without being willing to pay extra money for it. For functional safety the users have similar requirements.

In addition to the general challenges as listed at the end of the previous section there are further challenges dedicated to software security:

- The strength of the overall system relies on its weakest part.
- Software bugs are difficult to find in terms of time and effort. While it must be assumed that attackers can spend a lot of time, industry development engineers don't due the pressure of a fast time to market.
- For an attacker it may be sufficient to find a single bug that leads to an exploitable vulnerability.
- Attackers as well as the attacks themselves get smarter over time.

FIGURE 2 Passenger vehicle life-time cycle



As a result, attacks become more likely and possibly accompanied by more negative impact. Thus, applying security by design approach becomes essential and includes the application of established measures, such as "defense in depth". This is a layered security approach, which is composed out of multiple, independent layers of security. Due to the previously outlined complexity and the fact, that a potential attacker has physical access to the vehicle and unlimited amount of time implementing security in software only becomes a risk. Thus, software cannot protect software in this case. This raises the need for hardware-based security.

Overall Protection Goals

Typically, IT security goals are built on the three cornerstones of information security: confidentiality, integrity, availability ("CIA") (Figure 3). However, this is not sufficient in the context of automotive security, since a vehicle is not an IT system but needs to be perceived as a "cyber physical system" which also has real-world safety impact. Thus, the classical "CIA" approach misses important aspects and a holistic view is needed encompassing the following overall automotive security goals, which are expected from a secure vehicle:

- Provide functional safety
- Finance: Protect business and intellectual property (IP)
- Operations: Meet customer's quality expectation
- Fulfill privacy and regulatory requirements

These overall goals are to be applied to the several security pillars as depicted in Figure 4.

1. Protection of devices and ECUs
2. Secured vehicle networking
3. Secured architecture supporting the protection of the central gateway and domain isolation
4. Protection of the connected car, in particular by securing the externally accessible interfaces

The pillars are thereby intended to complement each other. Secured ECUs shape the basis for networks by following the defense-in-depth approach. Secured interconnection, in return, requires trusted devices, which then provide the fundament for secured communication.

FIGURE 3 Overall protection goals.

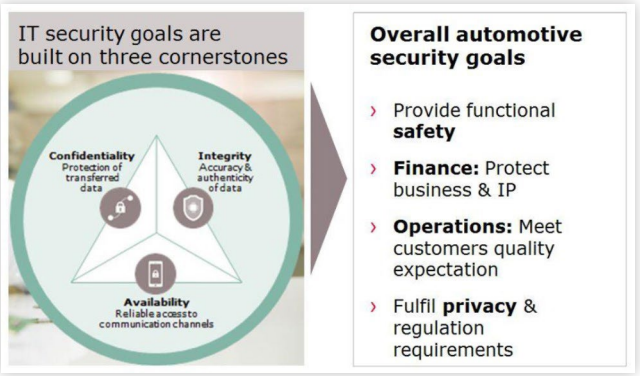
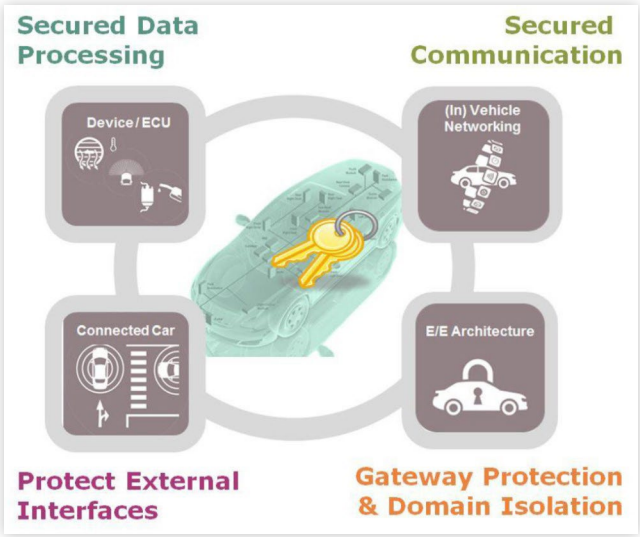


FIGURE 4 Pillars of Automotive Security



The Importance of Hardware Trust Anchors for the Protection of Cryptographic Keys

In the end, secure vehicle functions rely on cryptographic keys as a basis. Thus it is essential for the security of the system to protect the integrity and confidentiality of these keys i.e. once they are compromised or leaked the underlying security concept may be undermined. Read-out and cloning of keys leaves no traces, thus it is impossible to distinguish between intentionally distributed and stolen/duplicated keys later on. The management of keys has to be ensured throughout their life-cycle over the long automotive life-cycle (as depicted in the previous section). The life-cycle includes thereby the creation, distribution, deployment, usage and revocation of keys from production line to decommissioning of the corresponding ECU. If a key needs to be revoked it is timely and very costly for the car manufacturer.

Thus it is essential to implement strong measures for storing these keys in a vehicle which leads to the concept of a hardware-based trust anchor. It provides a tamper-resistant hardware and a protected runtime environment to achieve a high security level over lifetime. From this "root of trust", which stores the most valuable and sensitive keys, further keys or security measures can then be derived to realize secured vehicle functions (on the base of the trust anchor).

In particular, a hardware trust anchor allows hosting key storage, related cryptographic operations as well as key management and deployment in insecure environments. Referring to the previously outlined complexity issue, this means that the approach of a hardware trust anchor encapsulates the critical, security-related part and separates it from the larger and also complex main system. This results in a manageable small trusted computing base. Thus, it enables the underlying hardware to form a basis for secured vehicle functions.

The threat landscape and attack scenarios will change rapidly in the next years, requiring quick counter-measures in certain circumstances. Since the hardware is fixed and equipped to cars with up to 24 years life-cycle it is very difficult to perform a bug fix based on hardware for all existing cars in the field. Even for cars that are not produced yet, it will take some time until new secure hardware is developed, verified and implemented into the car production. Thus, immediate response concerning detected attacks must be done via software changes, such as adapting data and behavior (e.g. configurations, Firewall-ruleset, etc.). Furthermore cryptographic algorithms and/or encryption schemes have to be changed.

Although it is quite inflexible, hardware in the form of secure elements is a very important pillar in each security concept. Tamper-resistant hardware is required for critical assets (c.f. embedded attacks in the introduction chapter). Hardware accelerators are also necessary to meet the performance requirements. To fulfill both requirements a coexistence of software (to stay flexible over car live-cycle) and hardware (to form a solid, tamper resistant base) is necessary.

Different Protection Levels of Hardware

Standard semiconductor hardware devices can be attacked on different levels:

- Logical attacks: e.g. fuzzing.
- Monitoring attacks: e.g. analysis of power profile, probing of I/O signals.
- Semi-invasive attacks: e.g. side-channel analysis, laser fault injection.
- Manipulative attacks: e.g. probing on the silicon-die.

The effort of attacks increases from top to bottom and so does the necessary technical effort and the associated costs. The same applies to the defense against these types of attacks (c.f. Figure 5): The stronger the hardware attack, the more measures are required, i.e. the deeper a given attack targets the hardware, the more hardware is necessary for defense and vice versa. Thus, the higher the intended level of security level, the higher the need for hardware based measures.

For the attacks common today, which are presently mainly software-based, it is essential to have counter measures in hardware available. Preferably an appropriate combination of software and hardware countermeasures is used. Thereby the contribution of secure coding is partly still underestimated and includes, e.g. measures like code redundancy or randomization.

It is important to emphasize that hardware-based attacks are not restricted only to highly sophisticated attackers any longer but reach a broader accessibility, since both attackers and attacks get smarter over time.

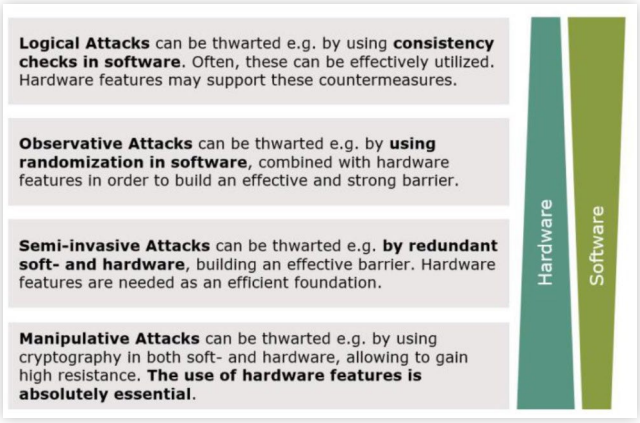
The technologies and equipment necessary for accomplishing certain attacks get better and cheaper and reached a level where assumptions are refuted that have been made just a couple of years ago. A good example is the claim that a given attack is not feasible due to the high cost, necessary technical skills and expertise and also the missing "opportunity" as e.g. an IMSI catcher today can just be emulated via software defined radio whereas it has formerly been a costly device. Well-known examples of such easy-to-use and freely available low-cost devices include Chip Whisperer [10] or HackRF [11] for software defined radio.

From an attacker perspective hardware-based attacks gain importance for several reasons:

- Attacks and equipment got cheaper, scale better and thus are easier to reproduce.
- (Detailed) information can be easily gathered.
- Hardware and software tools are available at low cost, which lowers the entry barrier.

As a result there is the need for a shift towards hardware based security measures, since OEMs have to continuously raise the security barrier over the next years.

FIGURE 5 Countermeasures overview: Opportunities and limits



Why Trust Anchor in Hardware and What Are the Benefits?

Without any security measures, software can easily be extracted and analyzed. Security mechanisms employed on a software level only allow the protection against casual software attacks whereas hardware based measures go beyond and provide protection against hardware attacks as well - c.f. Figure 6.

FIGURE 6 Benefits of hardware based security

From a security perspective hardware-based solutions provide strong physical protection by being tamper-resistant and thus enabling long lasting effectiveness. They provide protected processing, safe firmware storage and keeping processed data preserved in memory. This is enabled through encrypted memory and processing, fault and manipulation detection as well as secured code and data storage. Thus, software running on secured hardware is protected from reading, copying and cloning and also from being analyzed and sabotaged. To this end hardware-based security provides an important fundament for a rich set of functionalities that can be implemented on top of it. Utilizing discrete hardware offers unique benefits even beyond an increased level of security.

Security and Certification

Due to certification by 3rd parties trust anchors provide evaluated security creating higher trust in the security of the system. Certification provides additional assurance that the solution is secured and that this statement has been extensively tested and certified by an independent third party lab (e.g. a government agency). This is especially important as security testing (unlike normal product testing) aims at testing the hardware outside of common specifications because this is where attackers find their targets. To perform this approach the testing requires extensive knowledge and experience as well as many resources (e.g. manpower). However once a product has successfully run through this extensive process, there will be a much better base to select this product as a secured solution as there is external "proof" of the security level.

Time to Market and Secure Implementation

Implementing a secure solution without bugs requires a lot of experience. By using (certified) hardware solutions the risk of bugs can be significantly reduced as many years of experience were put into the design and development of the product. Discrete hardware-based security solutions hereby allow offloading the primary system from cryptographic tasks and thus reduce complexity. This reduces the time to market while also reaching the intended security level.

As a result, the saved development time can be spent to put more effort into the main product features and business cases while still complying with regulations.

Secure Implementation

There is one major difference between the implementation of a security feature compared to implementing other features. While for other features the implementation can be tested against known issues, the implementation of a security feature requires the protection against known attacks as well as the consideration of possible future attacks. This introduces additional aspects which have to be considered for the implementation of security features:

- Experience is required to implement a security feature correctly without bugs.
- Right implementation: By leveraging certified hardware, a solution based on many years of experience and, in addition, approved by external entities is provided.
- Complexity: In comparison to a normal implementation, security code aiming at side-channel resistant implementation typically doubles the code size [12]. This opens a wide range for faulty implementations, especially when no expertise in secure programming is at hand.

Cryptographic operations are an important aspect of security solutions. Operations (symmetric and asymmetric) are implemented hardware based in a dedicated module (e.g. co-processor) and/or in software as cryptographic libraries. This typically includes a true random number generator and anti-tearing (allowing the chip to always return to a secured state) as well as secured storage. All of these functionalities are used in a tamper resistant environment. On top of that they need to be securely implemented. This requires the corresponding expertise in secure coding.

Scalability and Innovation

Discrete hardware-based security solutions can easily scale across various ECUs using the same security model across the entire vehicle. In addition they provide the latest (certified) security innovations regardless of the life cycle of the other components of a vehicle platform.

Although ECUs aren't identical, the basic security requirements are very similar - i.e. safeguarding integrity, authentication, confidentiality and/or availability of the devices and the therein processed data. Therefore using the same implementations across several vehicle ECUs improves the scalability.

Compared with alternative technologies in implementing security, dedicated hardware trust anchors have strong advantages. For example a software security implementation has to be ported and verified on the above mentioned variety of microcontrollers which do offer different operating systems, different performance and different capabilities (and in particular different security level). Therefore every implementation is individual to the system.

In case of an integrated security implementation the hardware implementation of the security function in another microcontroller has to be available in all microcontrollers or CPUs used in that environment (car). It increases complexity and limits time-to-market.

Thus, a dedicated hardware can easily be deployed across a vehicle providing the same functionality in the same quality of implementation and with the same high security level.

Logistics and Costs of Ownership

Hardware-based security enables flexibility at manufacturing sites and reduces costs as it allows production in less secure environments. It also allows to verify where components were sourced (secure value chain).

Thus, it reduces the total cost of ownership for security based on the security savings resulting from hardware security

benefits. This includes, that no secured manufacturing and less investment in security know-how is needed.

Handling security devices in a supply chain and following all export regulations requires corresponding knowledge and experience. The development and production of secured devices and corresponding facilities have to provide certain security environments. The protection of secured products within the supply chain must be provided until it reaches a secured environment. By utilizing a hardware-based security device the complete work for the above is with the specialized manufacturer of the security chip.

Pre-personalization of secured hardware allows to track the components along the value chain without any additional efforts. This is outlined on the basis of the following example, which assumes, that the security concept is based on cryptographic keys which are used to authenticate the ECUs and services in a reliable way. A popular representative of such an approach are PKI infrastructures utilizing public-key cryptosystems, such as RSA and ECC - both requiring a key pair. A key pair consists of one public key which is known to everybody and one private key which must be protected and securely stored in the device itself. If the private key gets compromised, other devices (ECUs) can mimic the original device, thus damaging the integrity of the complete system. A critical step in this context is the download of the private key into the ECU. If done properly, the location where the download happens is physically protected (e.g. by using a separate room with access restrictions and monitoring). The computer system where the keys are managed must be well shielded against attacks from the outside. This process can be done within a specially secured environment of the manufacturer of the security chip which is dedicated to this purpose. After shipment of the semiconductor IC the private key cannot be read out. This is a big advantage, since it significantly reduces the manufacturing and logistics efforts on the side of car manufacturer and ECU supplier. Thus less investment in secure infrastructure for development and manufacturing is required, which in turn provides flexibility while reducing costs.

While hardware-based security might increase the BOM it has further advantages on costs - besides the previously outlined logistic advantages: Due to the thorough testing certified solutions have a lower risk to be compromised. This reduces the risk of replacement costs and loss of reputation. For example the setup of a secured production environment for just one product costs around 300-400 k EUR for security infrastructure and takes between six months and a year. In addition regular evaluations of this infrastructure must be done.

Privacy Protection

Discrete hardware-based security solutions, such as TPM, allow the anonymization of keys. To this end, a cryptographic primitive called "direct anonymous attestation (DAA)" is deployed. This scheme is adopted by the Trusted Computing Group (TCG) as the method for remote authentication of a Trusted Platform Module while preserving the privacy of the user of the platform that contains the module.

The utilized DAA protocol is based on three entities (i) the DAA member (i.e. TPM platform), (ii) the DAA issuer and (iii) the DAA verifier. Thereby the issuer verifies the platform and issues a credential to the platform. The platform (member) uses this credential with the verifier, who can verify the credential through a zero-knowledge proof without attempting to violate the platform's privacy.

DAA represents a group signature without the feature that a signature can be opened. That is, the anonymity is not revocable. Additionally, DAA enables pseudonyms, i.e. for each signature a user can decide whether the signature should be linkable to another signature. Furthermore DAA features the detection of "known keys". That is, in case the DAA secret keys are extracted from a TPM and subsequently published, a verifier can detect, that a given signature has been created using these secret keys. The scheme is thereby provably secure under the strong RSA and the decisional Diffie-Hellman assumption.

Performance

Hardware based security offers remarkable performance advantages compared to software-based solutions for secured storage and processing.

This refers not only to the performance of cryptographic operations but also to the implementation of countermeasures against certain hardware attacks, such as side-channel attacks. For example masking is a countermeasure that randomizes the intermediate values of cryptographic algorithms while the input and output of the cipher stay the same as in the unmasked version. The implementation of this countermeasure can be done in hardware and software.

For example, in case of AES encryption it may encompass inserting a random number of dummy data or shuffling the order of S-Boxes for every data block. Generally, implementation in hardware is thereby more resistant to power analysis attacks than a corresponding software-based solution [13]. Based on the example of an AES core implemented on an ASIC, researchers found in [14] that substantially more effort is necessary to attack the hardware compared to the corresponding software version. In order to securely hide the calculation done with a cryptographic key in terms of the changes in power consumption a tamper-resistant hardware needs significantly less masking calculations in order to securely protect the key compared to a software solution.

Survey on Hardware Security Devices for Automotive Applications

In this section we present hardware security devices for automotive applications that are available today and present the evolution of requirements in the automotive industry.

In the past, the reasons for security in the car were very simple. Nobody should be able to steal a car (anti-theft) and nobody without authorization should be able to open up a car to remove valuable items. In addition it should be very hard

even for specialists to tune the engine or manipulate the odometer. Since the car was a closed system with almost no external interfaces these requirements were manageable with "weak" solutions like SHE (Secure Hardware Extension) [15]. But requirements evolved as connected cars are getting common today. Starting with 2018, each new car in the EU needs to have a telematics unit to exchange data with the service provider. In the future, this channel will be used to exchange data with other cars. The 5G mobile standard for example will accelerate this development.

In the following we present details on several available modules - SHE (Secure Hardware Extension), HSM (Hardware Security Module) [16] and TPM (Trusted Platform Module) [17] - and give an outlook on solutions required in the future (2020+).

Hardware security devices are commonly known as trust anchors. Trust anchors are dedicated execution environments that host secured storage and processing of keys. The main purpose is to protect the keys and encryption capabilities. No attacker should get into the position to take possession of this sensible information. Three aspects differ from general silicon production.

1. Although trust anchor manufacturers are usually part of a bigger semiconductor manufacturer company they typically form an encapsulated organization inside the company. E.g. no employees of other departments have access to the offices of the trust anchor department, IT infrastructures are completely separated. They can be seen as a company inside the company.
2. A semiconductor consists of different layers to build up the needed transistors. These layers are generated with different masks in several chemical and physical processes. The layout is different to trust anchor layout, since it does not try to hide information. A secured semiconductor layout tries to hide the information which is stored inside.
3. The software in which the encryption mechanism and keys are stored requires secure coding to hide information by bluffing and disorienting the attacker. Negative effects on the overall execution time and code size are accepted.

A SIM (Subscriber Identification Module) card may serve as an example for a kind of trust anchor. The legitimate owner of the card should be able to use the card to make calls and send/retrieve data with his smartphone. But nobody should be able to get hold of the keys stored on the card to prevent cloning and abuse. Actually the SIM card was the first application which found its way into the car. Either as classical SIM cards as known from mobile phones or in form of an embedded SIM. eSIMs are programmable devices inside the car, which emulate a SIM card with the same security features.

In cars, applications running on a microcontroller should be able to communicate securely but should not be able to read out or modify keys. A weak kind of implementation for a trust anchor is to build a trust anchor software component as part of the operating system of a microcontroller. It has been shown that this kind of protection can be broken and can't be considered as state of the art. The "Heartbleed bug" can serve as an example of a weakly protected key storage. This attack affected many industries and revealed the difficulties to recover from attacks on keys [18], [19], [20]. It demonstrated a great benefit of separating the trust anchor function from the overall system (Figure 7). In some way trust anchors mitigate the complexity problem of security by separating critical security functions from the overall system.

The automotive industry was aware of this problem. In the HIS consortium (Hersteller Initiative Software) the first standard for tuning-protection and anti-theft mechanisms has been set. This standard is known as SHE (Secure Hardware Extension), which is a kind of fixed state machine inside automotive microcontrollers which, for example, perform chassis or engine control functions.

Unfortunately for new applications like eCall (Emergency Call) or SOTA (Software over the Air) [21] this approach is no longer flexible enough. This resulted in the development of so called HSMs (Hardware Security Modules). HSMs are separated programmable domains inside the package of a microcontroller. They are separated from the microcontroller by a Firewall. HSMs have their own CPU, RAM and ROM and offer hardware accelerators for security functions. The HSM Standard was developed by the EVITA project [22], [23].

Processing of keys is a security critical operation. Therefore in trust anchors special measures are implemented that harden these operations against attacks. With respect to functionality trust anchors have a substantially lower complexity as the overall function of the ECUs which they are protecting. Therefore they can be evaluated and tested with a thoroughness which would be nearly infeasible for the overall system because of required time and effort. In a way they resemble the "divide and conquer" principle which is commonplace in engineering to cope with complexity of systems.

OEMs are starting to classify data with respect to security relevance. This classification is also affecting the measures which are required and justified for protection of the related keys. In general, communication that can affect the security of the whole fleet of an OEM is regarded more critical than

FIGURE 7 Trust Anchors connected to Host CPU

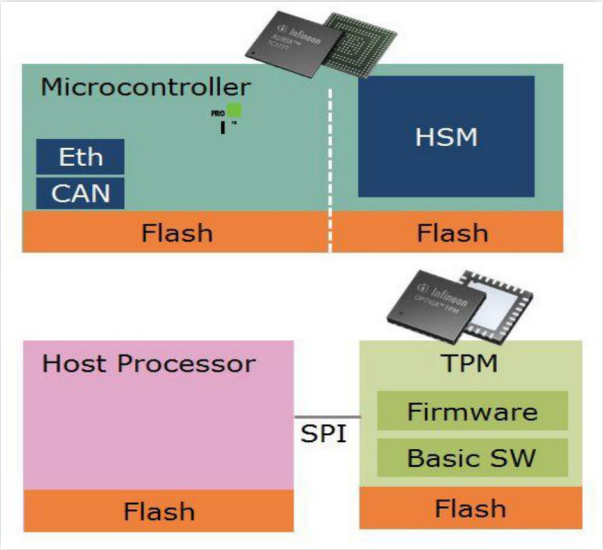
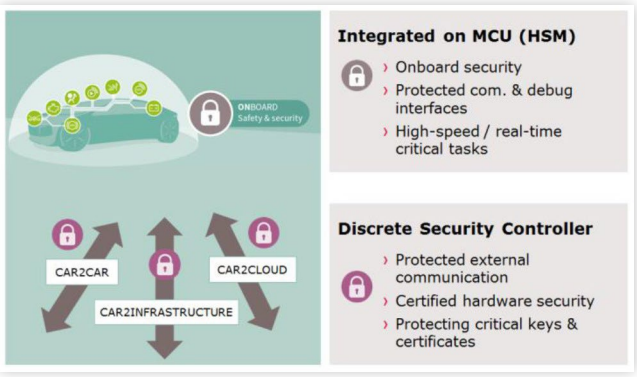


FIGURE 8 Classification of Trust Anchors used in vehicles



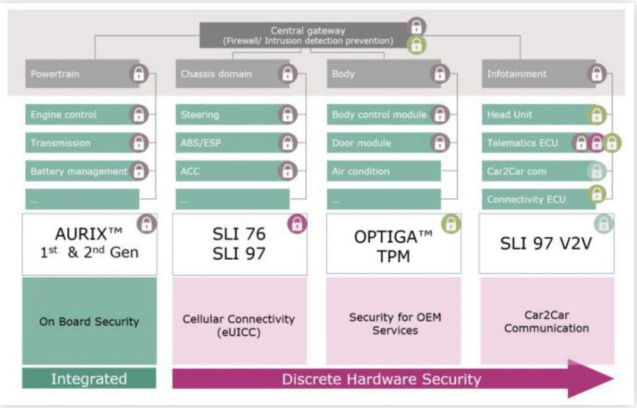
communication affecting only one specific car. Likewise the lifetime of a key is relevant. Long lasting keys should be protected stronger than keys that will only be used for a limited time (e.g. session keys).

In the automotive industry commonly two basic types of trust anchors are deployed, as depicted in Figure 8:

1. Trust anchors integrated silicon wise on automotive microcontrollers e.g. HSM
2. Trust anchors that are based on security controllers using smart card technology (e.g. Infineon OPTIGA™ TPM, NXP SmartMX and comparable).

The main use of the first category is secured on-board communication requiring high performance and good real time ability. The second category is used to secure external communication and can also serve as a central key storage for the car. In the latter category, Security Controllers include comprehensive hardware measures that protect them even against attacks using dedicated and sophisticated hardware and tools (e.g. side channel attacks by observing power consumption or even attacks on the silicon die). Additionally great care is taken to test and evaluate the critical software operations of those parts. Products are tested in many cases following a standardized security evaluation scheme called Common Criteria [24], [25] that is used widely in the smart card industry to successfully protect large, security critical infrastructures e.g. passports, signature cards, etc.

FIGURE 9 Automotive Trust Anchors



Aside the key storage trust anchors can host application specific algorithms. Vendors therefore offer various types of trust anchors which differ with respect to performance but also security. Using Infineon's product portfolio as an example, Figure 9 shows a range of security hardware solutions for different applications. Comparable products can also be found in the portfolio of other semiconductor vendors.

Overview about Trust Anchors

This chapter gives an overview about currently available types of trust anchors.

SHE

SHE is the acronym for Secure Hardware Extension. The feature set for SHE has been specified by HIS (Hersteller Initiative Software), which was a group of German Car Vendors. The realization was done in hardware as a state machine. This method made it quite secure, but very inflexible. It was (and is still) mainly used for the most urgent security topics like Immobilizing, Anti-Theft, Tuning-Protection, Secure Boot and last but not least as a key storage. To support this, it had a RAM for keys, a symmetric HW crypto engine and a true random number generator.

HSM

The HSM (Hardware Security Module) is an evolution of SHE in order to cover new security trends and to get rid of the inflexibility of the static SHE state machine. It represents an independent microcontroller (e.g. 32bit CPU with RAM, ROM and timer peripherals) in the same package as its host microcontroller but separated by an internal firewall. In general, a HSM also offers HW accelerators for performing cryptographic operations - like AES-128 encryption - and true random number generators. It is a highly flexible and programmable solution. Crypto- and algorithm- agility can be achieved by software. HSMs support one of the three levels of EVITA security classification.

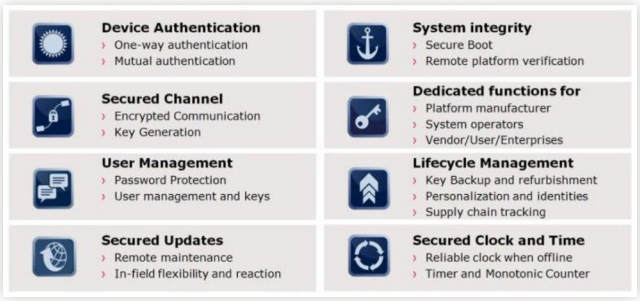
Software for a HSM can be obtained by the semiconductor vendors or security system houses. Big advantages using HSM are the flexibility, the high performance of the integrated HW accelerators and a fast interconnection to the Host CPU.

TCG TPM Standard

The TCG (Trusted Computing Group) is an organization whose objective is to enable secure computing with open standards and specifications. The latest specification is TPM (Trusted Platform Module) 2.0. The standard defines a powerful set of security mechanisms.

A TCG TPM IC is a standard security controller for cryptographic operations and tamper resistant key storage. The software which is implemented according to the TPM 2.0 standard should follow the rules of secure coding and its storage should be protected against unauthorized manipulation.

FIGURE 10 OPTIGA™ TPM security functions



A TPM is a passive device and physically separated from the main processor. Today, TPM ICs can be found in many computers and notebooks. Typically it is used by IT departments to store the key information for HDD or SSD encryption.

The device is capable to resist logical and physical attacks. To follow the common criteria standard, the security features are evaluated by a 3rd party.

Figure 10 shows some additional security features of OPTIGA™ TPM device as one example for an available standard product.

Chipcard

Chipcards arrived in the automotive environment with the introduction of services like eCall. One example for an automotive chipcard is the eUICC family from Infineon. Derivatives of this family offer hardware based cryptographic coprocessors supporting all relevant crypto schemes and Common Criteria certification up to EAL 5+. Since these devices are programmable the car manufacturer can easily deliver them with different software version for different baseband providers depending on the specific supply region.

TrustZone

TrustZones trusted execution environment runs on the same CPU as the rich operation system. It is a software based security solution. TrustZone is a valid implementation of a trust anchor when used in conjunction with an eFuse or similar technology. Hereby an eFuse is considered to be a set of registers which are used for storing device specific features such as device ID, etc. These device features are read-only and e-fused into the device. That is, it consists of one-time programmable bits embedded in hardware. Once set, these bits cannot be reverted. Thus, they can't be changed outside the factory of the chip manufacturer after production. TrustZone does not provide security for non-volatile storage and lacks a guaranteed root of trust.

Case Study: Function on Demand

Today we are more connected than ever. Smartphones, tablets and laptops allow the access to different services at our fingertips. The personal identity is shared between these devices. This also will apply to the vehicle. The automotive future is called "Function on Demand" and includes no less than the complete separation of functions and vehicle hardware. In other words, one day we will no longer buy our little electronic helpers for a special car, but for our personal car account [1].

The idea of "Function on Demand" creates unique advantages for car manufacturers and also vehicle end users. Car manufacturers can use vehicles within new business models and create greater customer satisfaction. A good example is the sale of updates for download. On one hand, this enables the OEM to collect information about user behavior and demand which in return introduces new possibilities for extending customer services as well as strengthening customer loyalty, also helping to establish a stronger bond to a car brand. On the other hand, the end customer can buy additional software functions (customer features) on demand. Acquired features could then either be available for a limited time period (e.g. for a weekend trip) or offered for unlimited use. Thereby the customer is no longer compelled to decide for functions which he may not need at the time of vehicle purchase. A further advantage is that the end customer can have access to new functions in the vehicle more quickly since new innovations are no longer strictly tied to the development or release cycles of vehicles. In the end, this allows the car owner or user to continually adapt the vehicle to his changing needs.

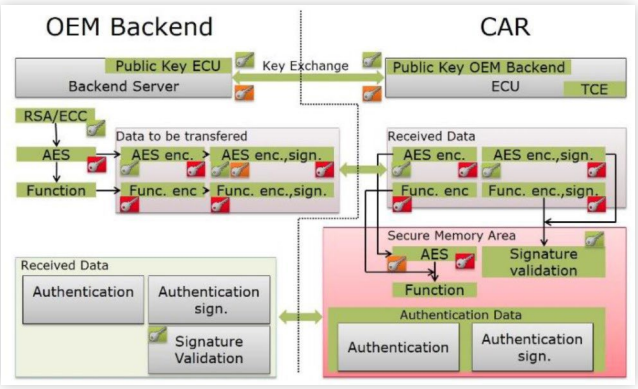
The main goal of all security measures is to maintain confidentiality, integrity and availability. This is still valid for the "Function on Demand" use case. The various communicating entities involved in such a scenario, e.g. a car communicating to the OEM back end but also ECUs within the car communicating to other ECUs, require that they can determine the authenticity of other entities. That means that an ECU must be able to verify that received data is not modified and originating from an authorized source. Additionally, it may be required to prevent eavesdropping of communicated data.

As the providers of new functionalities or automotive services want to protect their business models against unlicensed copies or unauthorized service usage, there is a justified need for effective countermeasures to prevent such scenarios. Since the business case depends on cryptographic keys for enabling to switch function on and off the required, key-infrastructure is a vital part of the model. Key material needs effective protection throughout the whole life cycle of the car, not only in the field but also during development and manufacturing processes already (Figure 11).

Using cryptographic algorithms and remote authentication via an OEM back end represent the first step to protect the business case as well as the customer asset. Furthermore, the key data used to compute the cryptographic algorithm in order to calculate the authorization to enable a specific function needs to be stored in a secured environment.

Unlocking of functions must be secured along the entire chain from the back end to the control unit. Since a potential attacker may even have physical access to the vehicle, the therein contained assets utilized for "Function on Demand" must be stored in a tamper-resistant way to achieve effective protection against attack attempts targeting the physical device and thus the hardware itself.

FIGURE 11 Basic concept for a key infrastructure



Therefore the following basic security requirements can be summarized:

- A complete end-to-end connectivity must be provided.
- Codes for enabling functions must be digitally signed by the IT back end and are transferred via TLS.
- Digital signing must be carried out based on an individual key per control unit.
- All key material as well as release codes must be stored in a tamper-resistant manner.
- After every change of the function (enable/disable) the information must be transferred to the back end for documentation and validation purposes.

Conclusions

Automotive security in a holistic approach is a big challenge for both the car manufacturer and the suppliers of secure elements. It is broadly accepted, that new car features like automated driving, Car-2-Infrastructure communication, etc., transforming a car into a smartphone on wheels, need a holistic security solution. It would be no good advice to decision makers at car manufactures to develop their own concepts from scratch. Concepts and methods from other industries like consumer electronics, banking and insurance sector can be utilized and adapted to the particular needs of automotive applications. These industries have long-term experience in handling of security-related processes as well as already standardized security mechanisms sets like TCG TPM.

However, automotive industry is different from consumer industry. The product life cycle in consumer industry is typically much shorter with about three years or even less. This short period makes it very easy to ignore the need for cryptographic agility.

Thinking about more than 20 years by adding development, production and maintenance time, it's not an option for car makers to bring a fixed security concept into their products. Instead, the flexibility for updates to cover new hacks and match with state-of-the-art security for the whole lifetime of a vehicle is required. Possible solutions also need to withstand the harder automotive environmental requirements.

In addition, it can be expected that future legislation and regulatory authorities, sooner or later, will no longer accept unrestricted operation of vehicles with known security weaknesses.

This paper showed the necessity of trust anchors within a holistic vehicle security concept. Furthermore it listed today's commonly used trust anchors with their features.

We came to the conclusion, that hardware (secure elements/trust anchors) and software (security algorithms) can't be separated when a maximum of cryptographic agility over the typical life cycle of passenger vehicles is intended.

Car manufacturer, component suppliers and security software vendors need to work closely together in order to develop hardware and software that fulfills security best practices. Similar to ISO 26262 for automotive safety, there is a strong demand for commonly accepted standards for automotive security - like ISO 21434 and SAE J3101, which are currently work in progress. Once published, they would also support a common language and thinking to simplify the way to good solutions.

References

1. Audi Digitalstrategie, http://www.spiegel.de/auto/aktuell/audi-digitalstrategie-extras-fuer-gewisse-stunden-a-1105990.html, accessed Feb. 2018.
2. Falliere, N., Murchu, L.O., and Chien, E., "W32.Stuxnet Dossier," White Paper, Symantec Corp., Security Response, 5(6), 2011.
3. Felt, A.P., Finifter, M., Chin, E., Hanna, S. et al., "A Survey of Mobile Malware in the Wild," in *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, ACM, October 2011, 3-14, doi:10.1145/2046614.2046618.
4. Gollmann, D., Gurikov, P., Isakov, A., Krotofil, M. et al., "Cyber-Physical Systems Security: Experimental Analysis of a Vinyl Acetate Monomer Plant," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, ACM, April 2015, 1-12, doi:10.1145/2732198.2732208.
5. Li, C., Raghunathan, A., and Jha, N.K., "Hijacking an Insulin Pump: Security Attacks and Defenses for a Diabetes Therapy System," in *2011 13th IEEE International Conference on e-Health Networking Applications and Services (Healthcom)*, IEEE, June 2011, 150-156, doi:10.1109/HEALTH.2011.6026732.
6. Luo, A., "Drones Hijacking - Multi-Dimensional Attack Vectors and Countermeasures," in *DEFCON 24*, 2016.
7. Checkoway, S., McCoy, D., Kantor, B., Anderson, D. et al., "Comprehensive Experimental Analyses of Automotive Attack Surfaces," in *USENIX Security Symposium*, August 2011.
8. Miller, C. and Valasek, C., "Remote Exploitation of an Unaltered Passenger Vehicle," Black Hat USA, 2015.
9. Million Lines of Code, http://www.informationisbeautiful.net/visualizations/million-lines-of-code, accessed Jan. 2018.
10. Chipwhisperer, https://newae.com/tools/chipwhisperer/, accessed Jan. 2018.
11. HackRF, http://greatscottgadgets.com/hackrf/, accessed Jan. 2018.
12. Hoheisel, A., "Side-Channel Analysis Resistant Implementation of AES on Automotive Processors," Master Thesis, Ruhr-University Bochum, June 2009.

13. Zhang, L., Vega, L., and Taylor, M., "Power Side Channels in Security ICs: Hardware Countermeasures," University of California, San Diego, CA, 2016.
14. Mangard, S., Oswald, E., and Popp, T., "*Power Analysis Attacks: Revealing the Secrets of Smart Cards*," . Vol. 31 (Springer Science & Business Media, 2008). ISBN:978-0-387-38162-6.
15. Escherich, R., Ledendecker, I., Schmal, C., Kuhls, B. et al., "SHE - Secure Hardware Extension - Functional Specification," Version 1.1, Hersteller Initiative Software (HIS) AK Security, Oct. 16, 2009.
16. "Introducing Hardware Security Modules to Embedded Systems," https://vector.com/portal/medien/cmc/events/Vector_EMOB_2017_Phanuel_Hieber.pdf, accessed Jan. 2018.
17. Arthur, W., Challener, D., and Goldmann, K., "*A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security*," First Edition (Apress, 2015). ISBN:978-1-4302-6584-9.
18. Heartbleed bug, https://en.wikipedia.org/wiki/Heartbleed, accessed Jan. 2018.
19. Trust Computing and Heartbleed, http://www.trustedcomputinggroup.org/avoiding-heartbleed/, accessed Jan. 2018.
20. Financial Industry Affected by Heartbleed, http://www.fsroundtable.org/financial-services-industry-swats-heartbleed-bug/, accessed Jan. 2018.
21. Steurich, B., Scheibert, K., Freiwald, A., and Klimke, M., "Feasibility Study for a Secure and Seamless Integration of over the Air Software Update Capability in an Advanced Board Net Architecture," SAE Technical Paper 2016-01-0056, 2016, doi:10.4271/2016-01-0056.
22. EU-Funded Project (2008-2011) on Secure Automotive Onboard Networks, www.evita-project.org, accessed Jan. 2018.
23. Weyl, B., Wolf, M., Zweers, F., Gendrullis, T. et al., "Secure On-Board Architecture Specification," *EVITA Deliverable* D3(2), Aug. 2011.
24. Common Criteria Main Page, https://www.commoncriteriaportal.org/, accessed Jan. 2018.
25. Lomne, V., "Common Criteria Certifications of a Smartcard: A Technical Overview," in *CHES 2016*, Santa Barbara, CA, 2016.

Contact Information

Christopher Corbett
AUDI AG
85045 Ingolstadt
Germany
christopher.corbett@audi.de

Martin Brunner
Infineon Technologies AG
85579 Neubiberg
Germany
martin.brunner2@infineon.com

Karsten Schmidt
AUDI AG
85045 Ingolstadt
Germany
karsten.schmidt@audi.de

Rolf Michael Schneider
AUDI AG
85045 Ingolstadt
Germany
rolf.schneider@audi.de

Udo Dannebaum
Infineon Technologies AG
85579 Neubiberg
Germany
udo.dannebaum@infineon.com

Definitions/Abbreviations

AES - Advanced Encryption Standard
BOM - Bill of Material
CAN - Controller Area Network
CRC - Cyclic Redundancy Check
ECC - Elliptic Curve Cryptography
ECU - Electronic Control Unit
eSIM - Embedded Subscriber Identification Module
FBL - FLASH Boot Loader
HDD - Hard Disc Drive
HSM - Hardware Security Module
IMSI - International Mobile Subscriber Identity
MCU - Micro Control Unit
MILS - Multiple Independent Layer of Security
OEM - Original Equipment Manufacturer
RSA - Rivest-Shamir-Adleman cryptosystem
SHA - Secure Hash Algorithm
SHE - Secure Hardware Extension
SIM - Subscriber Identification Module
SOTA - Software Over The Air
SPI - Serial Peripheral Interface
SSD - Solid State Drive
TLS - Transport Layer Security
TCG - Trusted Computing Group
TPM - Trusted Platform Module

ISSN 0148-7191