Security and Safety Co-Engineering of the FlexRay Bus in Vehicular Networks

Dominik Püllen*
Technische Universität Darmstadt
puellen@seceng.informatik.tu-darmstadt.de

Nikolaos Athanasios Anagnostopoulos*
Technische Universität Darmstadt
anagnostopoulos@seceng.informatik.tu-darmstadt.de

Tolga Arul*
Technische Universität Darmstadt
arul@seceng.informatik.tu-darmstadt.de

Stefan Katzenbeisser*
Technische Universität Darmstadt
katzenbeisser@seceng.informatik.tu-darmstadt.de

ABSTRACT

Recently, researchers have demonstrated how the lack of security features in road vehicles may allow adversaries to take over partial or even full control. Specifically, in-vehicle communication protocols are prone to attacks, because no security mechanisms have been developed for them. For a long time, they have been optimized only towards safety, in order to guarantee a high degree of reliability, robustness and real-time behavior. In this work, we focus on FlexRay, an automotive communication protocol whose core properties are strong determinism and high fault-tolerance for safety-critical applications. We propose to leverage the distinct safety-tailored features of FlexRay for security purposes, such as authentication. In particular, we demonstrate that the optional dual-channel mode can be used to provide authentication for FlexRay in a backward compatible manner. Additionally, we suggest different ways of transmitting the relevant message authentication codes over FlexRay. Finally, we propose a number of techniques for managing cryptographic keys, i.e. we associate these keys with FlexRay time slots and we use hash chains to derive new keys at low cost at runtime. In this way, we offer multiple security solutions for the FlexRay protocol, while trying to keep the overhead low.

CCS CONCEPTS

• **Security and privacy → Authentication**; *Security protocols*; • **Computer systems organization → Real-time systems**;

KEYWORDS

FlexRay, security, safety, fault-tolerance, reliability, automotive, in-vehicle communication

ACM Reference Format:
Dominik Püllen, Nikolaos Athanasios Anagnostopoulos, Tolga Arul, and Stefan Katzenbeisser. 2019. Security and Safety Co-Engineering of the FlexRay Bus in Vehicular Networks. In *INTERNATIONAL CONFERENCE ON OMNI-LAYER INTELLIGENT SYSTEMS (COINS), May 5–7, 2019, Crete, Greece.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3312614.3312626

* All authors are members of the Security Engineering Group of the Technical University of Darmstadt, which is located at Mornewegstraße 32, 64293 Darmstadt, Germany.

COINS, May 5–7, 2019, Crete, Greece

ACM ISBN 978-1-4503-6640-3/19/05…$15.00
https://doi.org/10.1145/3312614.3312626

1 INTRODUCTION

In recent years, the security of in-vehicle communication has received increasing attention, since researchers demonstrated how road vehicles may be targeted by hackers [8, 13, 15]. For decades, automotive engineers put a strong focus only on safety in the design process of in-vehicle communication protocols. Reliability, fault-tolerance, and robustness had the highest priority, while security was neglected. However, in the future, road vehicles will be intelligent and autonomous systems of systems with an increased interaction with its environment. Therefore, security has become an essential requirement. Additionally, vehicles already incorporate a complex network of sensors and actuators, which are intended to function as an autonomous system in the future, resembling an in-vehicle internet of things. Thus, apart from external communications, in-vehicle networks also need to be adequately protected, in order to ensure a high level of safety.

LIN [2], CAN [1], FlexRay [3], and MOST [4] are the most common in-vehicle communication protocols of contemporary vehicles. Each of them targets a specific use case, such as the activation of small, non-critical actuators, like window openers (LIN), general-purpose and prioritized communication (CAN), strongly deterministic and fault-tolerant networks for highly safety-critical applications (FlexRay) or the large bandwidth required for infotainment systems (MOST). Additionally, in order to handle significantly larger bandwidths, automotive manufacturers now strive for Ethernet-based systems. However, although Ethernet is cheap, well-known and comes along with many security solutions, it is a generic protocol that does not inherently incorporate the safety features of existing automotive protocols, such as FlexRay.

Vehicles have traditionally been considered a closed system with hardly any connections to the outside world. Over the last few years, however, road vehicles have been equipped with an increasing number of interfaces, such as Bluetooth, WiFi, and LTE. In addition, Electronic Control Units (ECUs) are supposed to receive over-the-air updates, which open new attack vectors. Adversaries may exploit vulnerabilities to gain access to in-vehicle buses. In case such buses are not well secured, counterfeit commands may pose a severe threat to the safety of both passengers and their environment. Vehicles are becoming ever more transparent and susceptible to cyberattacks, due to changes in their nature, such as the prevalence of smart interconnected vehicles and autonomous driving. This is why, we need security and safety co-engineering processes for today's automotive communication.

Most often, in-vehicle communication is realized by using multiple protocols for different components of the same vehicle. Therefore, to protect in-vehicle communication, *all* protocols used have to be secured. So far, researchers have proposed many solutions for the *Controller Area Network* (CAN) [10, 12, 17]. Additionally, Ethernet provides many security concepts, such as VLAN, 802.1X, VPNs, and IPSec. However, FlexRay, a deterministic and fault-tolerant protocol for highly safety-critical automotive applications, has been left out.

Contributions of this work:

In this work, we focus on security and safety co-engineering in the domain of in-vehicle networks and, more particularly, in regard to the FlexRay bus. In order to enable security for safety-critical applications in vehicular networks that utilize the FlexRay bus, we make the following contributions:

- Key Management: We suggest to associate cryptographic keys with FlexRay communication time slots. Moreover, we propose three techniques regarding the organization of cryptographic keys when securing the FlexRay communication bus. We also discuss how to define a session in a FlexRay network. Finally, we propose to use hash chains to lower the network overhead at runtime.
- MAC Transmission: We discuss different options how to transmit message authentication codes over a FlexRay channel. We differentiate between storing each MAC in an own static slot and integrating it to the frame that it authenticates.
- Dual-Channel Mode for Security Application: We propose to exploit the optional dual-channel mode of FlexRay, in order to transmit a message authentication code in parallel. A MAC is split into two parts and each is sent over a FlexRay channel. In case one channel breaks, the recipient is still able to authenticate the message.

To the best of our knowledge, this work is the first, that attempts to secure the FlexRay protocol. Additionally, the proposed schemes do not require changes in the manufacturing process of the FlexRay bus

Paper Overview:

The rest of this paper is organized as follows. Section 2 gives insight into related work. In section 3, we explain the FlexRay protocol and argue why it will still be relevant in future. Section 4 presents our main contributions, followed by a discussion regarding them in section 5. Finally, we conclude in section 6 and give an outlook about future work.

2 RELATED WORK

Researchers proposed multiple security protocols for automotive communication to prevent cyberattacks on modern vehicles. The main focus, however, has been placed on the Controller Area Network (CAN) protocol. The main challenges were backward compatibility, overhead reduction, and protocol integration. In this section, we briefly discuss some selected recent works, noting that little has been published about the security of the FlexRay protocol.

In 2015, Vasile et al. [19] compared different key distribution schemes for ECUs. They differentiated between four keying paradigms: 1) a key for all ECUs, 2) a key for each pair of ECUs, 3) a key for a group of ECUs, and 4) time-delayed keying. These four options can be used to address the trade-off between security and overhead. Key maintenance and exchange are easy for a single key. However, it suffices to compromise a single ECU to break the entire security of the vehicle system. In contrast, pairwise keying requires a key for all ECU pairs. Nevertheless, depending on the network size, this possibly yields a large overhead during the key exchange. In case the key arrives delayed, real-time behavior is more difficult to achieve. The authors conclude that group-keying should be used for in-vehicle networks, as it provides the best trade-off between overhead and security level.

Mousa et al. [14] adapted LCAP [12] to extend FlexRay with authentication, in 2016. LCAP has been initially proposed for CAN. In comparison to CAN, message authentication codes do not need to be truncated, because the FlexRay payload is larger. Data frames are not only authenticated but encrypted as well. However, LCAP suffers from a high number of messages that need to be exchanged in advance. During the setup phase, each pair of nodes has to exchange four control messages. Mousa et al. showed that LCAP does not require many modifications in order to use it for FlexRay, although it is not optimized for FlexRay. Specific FlexRay characteristics such as the periodic communication cycles or the optional dual-channel mode have not been taken into account.

In 2016, Nürnberger and Rossow proposed vatiCAN [16], a backward compatible framework for CAN message authentication. Messages are authenticated by computing a fresh HMAC and are transmitted as a CAN frame. The authors claim to deliver real-time protection by inducing a latency by 3.3 ms which results in an increased braking distance of 0.9 m at a speed of 100 km per hour.

In 2017, Fassak et al. [10] used elliptic curve cryptography for a session key generation between ECUs. A Diffie-Hellman key exchange sets up cryptographic keys on ECUs that are connected over the CAN bus. The idea is to append various MACs to support multiple receivers. However, this requires senders to know the recipients, which is usually not the case in broadcast protocols. The final MAC is truncated to 8 bits, so that it fits the CAN payload. Although they succeed in reducing the load, in comparison to other authentication schemes, such short MACs do not meet current security requirements.

3 THE FLEXRAY PROTOCOL

FlexRay [3] is a time-triggered in-vehicle communication protocol. FlexRay has been under constant development for more than fifteen years and is now defined by ISO standard 17458 [5]. It is designed for automotive networks requiring strong real-time behavior and used for highly safety-critical applications such as drive-by-wire, the active cruise control (ACC) and the anti-lock braking system (ABS). In contrast to its predecessor CAN, it provides higher data throughput, more deterministic behavior and a larger degree of reliability. It offers a maximum bandwidth of 10 Mbit/s on a single channel and a payload size of 254 Bytes. A FlexRay frame consists of a *header*, the *payload* and the *trailer*. The header is used for organizational tasks (e.g. synchronization) as well as message identification and is protected by a 11-bit error correction field. An

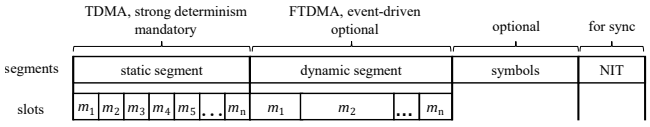

Figure 1: Structure of a FlexRay communication cycle

additional 24-bit CRC field that contains an error correction code for the actual payload, is appended as the trailer.

FlexRay communication is divided into periodic cycles. All cycles conform to a strict *communication schedule*. A cycle consists of a static and a dynamic segment. Static segments are divided into time slots that are assigned to network nodes in a TDMA fashion. Static slots are all of the same length and cannot be left out, i.e. they are mandatory. A maximum number of 1023 static slots can be used. By doing so, strong deterministic behavior and fixed latencies are guaranteed, which make FlexRay unique among the automotive communication protocols. A dynamic segment is optional, but of a predetermined length and likewise composed of slots, which are assigned to FlexRay nodes. However, once a node has the permission to transmit on a dynamic segment, it suppresses the transmission of other nodes, until the initial data transfer ends. Thus, prioritization is possible by allowing a specific node to start communication earlier in a dynamic segment. High-priority data pushes off low-priority data in dynamic segments. The transmission duration is only limited by the boundary of the dynamic segment. Dynamic segments are mostly used for low-priority data. The existence of static and dynamic segments combines strong determinism with event-triggered broadcasting, which is a well-known feature of CAN. In this way, FlexRay attempts to leverage the advantages of CAN and to address its shortcomings, such as its lack of determinism. Figure 1 shows how a communication cycle is organized.

FlexRay nodes need to be synchronized, in order to enforce the communication schedule. Bus guardians make sure that FlexRay nodes put data on the bus according to the time schedule. Each communication cycle roughly lasts 1 millisecond and is composed of macroticks which in turn consist of microticks. A microtick may consume a slightly different amount of time on different FlexRay nodes because each relies on its own crystal oscillators with possibly divergent behavior. The goal is to get a synchronized macrotick which typically lasts 1 microsecond.

The FlexRay protocol provides an optional redundant channel layout, in order to address highly safety-critical applications. Communication becomes more fault-tolerant due to the second, independent channel, over which the same data can be sent in parallel. This channel may instead be used to double the bandwidth being offered, to 20 Mbit/s.

4 SECURING THE FLEXRAY BUS

FlexRay is deployed for highly safety-critical automotive applications, such as drive-by-wire or powertrain communication, most notably in premium road vehicles. It offers an optional dual-channel mode which allows redundant data transmission.

In this section, we use data authentication as a prominent use case to secure in-vehicle communication. However, other cryptographic data such as certificates can be treated in a similar way, in order to transmit them over FlexRay. We first describe our attacker model. Then, we proceed to discuss the management of cryptographic keys, i.e. which key strategy should be adopted, as this has been an issue in other works [16, 18, 20]. We present ways in which cryptographic data can be integrated into the FlexRay communication and, finally, investigate the advantages of utilizing the dual-channel mode for security purposes.

4.1 Attacker Model

We assume an attacker who has full control over the network, but is not able to break cryptographic primitives. He can connect directly or remotely to the FlexRay network, monitor, record and publish messages (Dolev-Yao model [9]) model. Such an adversary is able to perform man-in-the-middle attacks, i.e. to intercept and drop messages. Authentication and decryption is only possible for the adversary if he is in possession of the correct key. We further assume that the attacker can compromise the ECU software. However, he is not able to access securely stored keys. In practice, a FlexRay network is not equipped with directly accessible interfaces, but it is connected to other network protocols over gateways. Such gateways can be exploited to eventually gain access to the bus.

4.2 Management of Cryptographic Keys

Cryptographic keys are a fundamental prerequisite for both encryption and authentication. Recent work has organized symmetric keys in two main fashions. Either a single key for a group of network nodes is issued [12] or keys are organized according to message types [18], i.e. message identifiers are associated with a key. The first approach scales linearly with the number of network participants, whereas the second one correlates with the number of message identifiers. We discourage the use of recipient-based keying techniques, because a sender usually does not know its recipients in a broadcast protocol. Instead of creating recipient-based or message-based keys, we propose to associate cryptographic keys with FlexRay time slots. The advantage of this method is better scalability regarding the number of network nodes and message types. We differentiate between three possible options:

(1) One key for all time slots.
(2) One key for each time slot.
(3) One key for a group of time slots.

The first option is obviously the easiest but less secure one. All nodes share the same key for all security operations. If the key is compromised, an attacker gains full control. Alternatively, each FlexRay time slot is associated with a key. As a consequence, all nodes that are interested in a specific slot require the corresponding key. The communication schedule is predefined and therefore, we argue that key establishment can be done during network setup time or dynamically within a key establishment phase. As a maximum number of 1023 static slots is theoretically possible, a FlexRay node would still need to administer 1023 keys, in the worst case. This may be infeasible in practice. Therefore, we recommend the third option which groups time slots in such a way that fewer cryptographic keys are required. In this way, the best trade-off between the number of keys and the security level can be achieved. In particular, if a certain FlexRay node occupies multiple successive slots, it may be

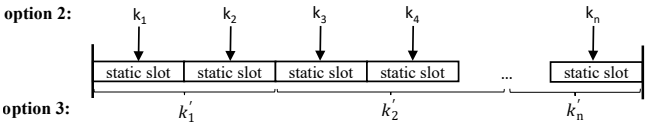

Figure 2: Keys are associated to time slots. Either a unique key is created for each slot (option 2) or for a series of slots (option 3). The actual choice depends on the network setup.

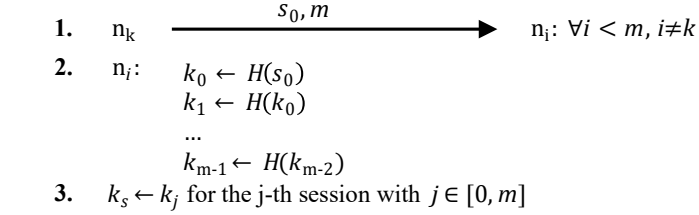

Figure 3: Each node n_i derives a chain of m cryptographic keys from an initial seed s_0 that is distributed during setup. The current last chain element k_j is the used as session key k_s.

reasonable to use the same key. Figure 2 illustrates the second and third option of key management, as discussed.

Cryptographic keys often have a limited lifetime. For this reason, [10, 12, 18, 20] use sessions for a secure CAN bus. However, the duration of a session has to be defined in advance. We differentiate between event-driven, time-driven, and load-driven sessions for FlexRay. The first type initiates a new session after the occurrence of a specific event. For instance, once the engine starts, all keys are renewed. The second option exploits the synchronization required by FlexRay between the nodes. After a specific amount of macroticks, a new session is started automatically and keys are renewed. The last option starts a new session once the network load is below a threshold. These three options can be combined in order to meet individual requirements.

Automotive sessions usually start with fresh cryptographic keys. In order to guarantee real-time behavior and lower the overhead induced by key exchanges, pre-computed key chains can significantly lower the expenses at runtime. A similar concept has been used by [12] to exchange magic numbers for message authentication. Instead of single and fixed keys for a specific number of FlexRay slots, we propose computing key chains based on the evaluation of secure hash functions. In this way, keys do not need to be distributed over the FlexRay network, each time a new key is due. Instead, new keys are derived from the key chain. Either after a predefined number of cycles or simply each time at startup, a fresh key is chosen. Recall that FlexRay nodes are naturally synchronized in order to adhere to the communication schedule. A hash-based key chain guarantees that adversaries do not learn anything about subsequent keys even if a specific key has been revealed. The computation of a new fresh key is performed locally on the FlexRay nodes. Thus, fresh symmetric keys can be derived without additional network traffic. The first key in use is the last chain element. Each subsequent key is the predecessor of the current one. For this, an initial setup is necessary to establish a key chain of length m on each FlexRay node. To do so, a secret seed s_0 has to be distributed among all recipients over a secure channel. Such a distribution can be initiated by a dedicated node. All other nodes subsequently conduct m recursive hash operations on s_0. Figure 3 illustrates these steps.

Let n_k be the initiating node. It has to be appointed in advance or dynamically during runtime. n_k sends a uniformly and randomly selected seed over a secure channel to all the other FlexRay nodes. The establishment of such a secure channel can be performed using asymmetric cryptography. The length of the key chain is denoted by m and is sent within the same initial message. Upon reception each node computes m keys and stores them securely.

The first key in use is k_{m-1}, i.e. the last element of the key chain. A fresh key is selected by moving back one step in the hash chain. The main advantage of this approach is that no additional communication between the nodes is necessary as long as new keys can be derived from the hash chain. Hence, no network overhead is added during runtime. Once all m keys are consumed, a new hash chain has to be established between the nodes. We suggest to use dynamic segments for security-relevant traffic. A master node (e.g. a trusted ECU) needs the highest priority to initiate the computation of a fresh key chain.

4.3 Transmission of Cryptographic Data

Data authenticity is usually achieved by computing *Message Authentication Codes* (MACs). A key problem in securing in-vehicle networks is the transmission of such additional cryptographic data, because there is no dedicated frame field for them. Additionally, the frame payload is relatively small. For instance, a CAN frame has a total payload of 8 Bytes and a FlexRay frame may store up to 254 Bytes. The Automotive Open System Architecture (AUTOSAR) [6] proposes to truncate MACs in order to squeeze them into a frame. This has indeed been done in prior works, e.g. [10, 12]. However, truncation decreases the security level, because the attacker's probability to brute-force a MAC increases. We present three options to transmit message authentication codes (MACs) over FlexRay channels. They differ in the timing overhead, the payload occupancy and their effects on the safety level. Depending on what is most important, the network designer can choose one of them or even combine these options accordingly.

The first option is the integration of a message authentication code into the frame that is being authenticated. This can be achieved by either appending it to the payload or by using other frame fields. The naive way to append MACs to the payload results in the reduction of the actual bandwidth, as less payload can be transferred at the same time. A 64-bit MAC would reduce the FlexRay transmission by roughly 3.15%. Recall that FlexRay frames only have a 254-Byte long payload. A way to occupy less payload is the exploitation of other frame fields, such as the *error correction fields*. Each FlexRay frame has 35 bits reserved for error correction. 11 bits are used for the header, while the trailer contains again 24 bits to protect the payload. Cryptographic data may be split and distributed across the two error correction fields. In case of 64-bit MACs, only 29 bits have to be stored in the payload. Hence, only 1.4% of the payload is used for authentication. However, error correction fields can only be used in case no backward compatibility is

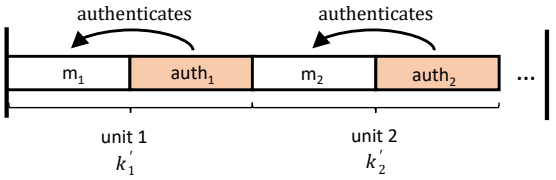

Figure 4: Transmission of cryptographic data under option 2: Authentication data are stored in an individual time slot. Here a unit consists of two slots.

required. Legacy systems will likely not be able to interpret frame fields differently.

The second, more convenient, FlexRay-specific option is the transmission of message authentication codes over an additional time slot. Two adjoining (static) slots are used to first transmit the actual payload and then the necessary authentication data. This is illustrated in Figure 4. This solution is fully backward compatible, as it does not affect the slot being used for payload transmission at all, therefore not requiring a new frame interpretation. However, the payload throughput decreases and the latency increases, because a FlexRay node has to wait for the authentication data in the next frame. Delayed authentication has been discussed for the CAN bus in [11]. This approach should be considered if the timing overhead does not lead to safety issues. A possible impact is discussed in the next section.

As stated in Section 3, the FlexRay protocol offers an optional dual-channel mode to either double the bandwidth or to transmit data redundantly over both channels. This yields us to the third option, which may be combined with the previous ones. The idea is to first compute a message authentication code *mac* and to subsequently divide it into two parts mac_1 and mac_2, where $mac = mac_1||mac_2$. According to AUTOSAR's technical recommendation, each part can be interpreted as a truncated MAC. However, instead of transmitting only one part to save payload, we send both in parallel using the FlexRay dual-channel mode. For instance, mac_1 is sent over channel A and mac_2 over channel B. This approach is illustrated in Figure 5.

We now face three possible situations.

(1) Both channels work properly. The transferred data can be authenticated by verifying $mac = mac_1||mac_2$.
(2) One channel breaks or gets compromised. The transferred data can still be authenticated using the truncated mac_1 or mac_2 respectively. Even though the full MAC is not available anymore, but only its most or least significant bits, authenticity can still be guaranteed in case *mac* is sufficiently large.
(3) Both channels fail. This scenario equals cut wires and cannot be handled with our methods.

The integration of mac_1 and mac_2 in the FlexRay frame on each channel can happen according to the first or the second option, i.e. in the payload and/or in the error correction fields, or in a separate time slot. In case the network designer decides to integrate them solely in the error correction fields, a 64-bit message authentication code can be transmitted without affecting the payload at all, as 32 bits can be embedded easily in these fields on each channel. While

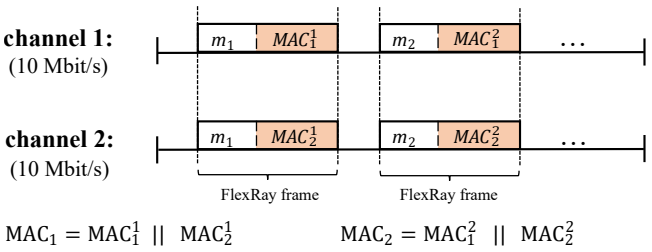

Figure 5: Transmission of cryptographic data under option 3: The message authentication code is split into two truncated parts, which are then separately sent over an individual channel.

the dual-channel mode was initially designed only for safety reasons, our work utilizes it in order to implement a security concept for the FlexRay protocol.

5 DISCUSSION

In this section, we discuss our proposed techniques to extend the automotive FlexRay protocol with security. At first, we concentrate on how data authenticity can be implemented in practice on FlexRay. In specific, we describe prerequisites for message authentication codes. Subsequently, we discuss the different techniques to transmit additional cryptographic data such as MACs over FlexRay channels. We balance the advantages of the different options regarding latency, backward compatibility and safety. Finally, we debate about FlexRay's future in the modern road vehicle, particularly because of the raise of Ethernet-based solutions.

5.1 Security Analysis

As a prerequisite, we assume the unforgeability of MACs under a chosen message (*UF-CMA*). In other words, the possession of valid Message-MAC pairs does not enable an attacker to create valid MACs for new messages. Nobody is capable to authenticate data without being in possession of a valid key. Consequently, fake message will eventually be detected and not be processed. Another important condition is freshness. According to our attacker model, the freshness of authentication data has to be guaranteed, in order to avoid replay attacks. This can be achieved by including a nonce into the authenticity computation. A message authentication code may either be based on hash functions (e.g. HMAC, VMAC, UMAC) or on a secret generator polynomial. In the latter case, agreement on a polynomial has to be reached during setup. The FlexRay protocol uses a publicly known polynomial for its error correction computations.

To allow fast key changes at runtime, we suggest to use precomputed hash chains. Their security primarily depends on the hash function's collision resistance. If a key gets stolen, an adversary cannot infer other keys as long as it is hard to compute pre-images of the hash function. Additionally, FlexRay nodes need to have efficient implementations of secure hash functions. Furthermore, in a real automotive network, secure storage for the cryptographic keys is needed by each FlexRay node. Moreover, we propose to assign cryptographic keys to FlexRay time slots. Even if one key gets compromised, the communication in the other slots remains

secure. Hash chains allow fast key changes at runtime without imposing a significant overhead. In all cases, an initial key exchange is necessary. Additionally, the deployment of a key hash chain requires to distribute the initial seed securely. This can be achieved using dynamic segments of a FlexRay cycle. The last key of a hash chain, i.e. the first element, can be used to encrypt a new, secret seed to the corresponding recipients over dynamic segments to establish a new chain.

5.2 Analysis of FlexRay Transmission

We describe different ways to transmit cryptographic data, such as MACs, over FlexRay. The first one embeds a MAC into the FlexRay frame itself. Either it is appended to the payload or stored into the error correction fields. We suggest using the first alternative, because only 3.15% of the total payload is occupied when embedding a 64-bit MAC into the 254-Byte payload field. Additionally, backward compatibility is maintained. However, in case the full-frame payload is needed, a 64-bit MAC can be distributed over the error correction fields of the FlexRay frame. By doing so, it can be split and transmitted partially over both FlexRay channels. Then, no frame payload at all needs to be occupied by the MAC, as each channel has to transmit 32 bits and a FlexRay frame incorporates 35 error correction bits. However, this choice should still be avoided, because the error correction function gets lost.

The second way to transmit MACs over FlexRay is to store them in an additional static slot. This induces a short time delay, because a FlexRay node has to wait for an additional time slot, in order to verify its previous one. However, once each second slot is used for authentication, the number of available slots is reduced by factor of 2. As, in total, there is a maximum of 1023 slots in each FlexRay static segment, the slots available to carry a true payload will be reduced to 511. Nevertheless, this approach is reasonable, as a similar work [16] proves. In that work, researchers send authentication data over an additional CAN frame, due to the strongly limited CAN payload. Even though bandwidth is reduced, this seems to be an appropriate way. As a FlexRay frame stores up to 254 Bytes, most payload would be unused if every second frame contains only a MAC. Therefore, in order to optimize the payload usage, the remaining space should be filled up with data of the next frame. However, the network designer has to take into account, that this is only possible in case the next frame still belongs to the same sending node. Recall that FlexRay follows a tight communication schedule and assigns slots to network nodes. Another side effect is an increase in latency, because each FlexRay node has to wait for two slots to process the content of one. Assuming a static slot lasts n microseconds, then, $2n$ microseconds are necessary to both process the message and verify its authenticity. If FlexRay implements a brake-by-wire system, the vehicles' brakes need to wait one additional static slot until a brake command can be verified and processed. This results in an increase of the reaction pathway. Assuming a road vehicle follows a uniform motion at 27.7 m/s and an extension of the reaction pathway below 0.01 m is acceptable, the additional time overhead must be below $\frac{0.01\,m}{27.7\,m/s} = 0.36\,\mu s$ according to the path-time law. This is a reasonable dimension which has to be taken into consideration during network setup, as the FlexRay network configuration allows to define the length of a cycle, the length of static segments and the duration of macroticks.

Finally, we suggest to use the dual-channel mode to transmit previously split authentication data in parallel over both FlexRay channels. This technique can be combined with the previous ones, e.g. to send MACs in separate slots over two channels.

If one channel breaks, messages can still be transmitted and verified. This is possible, because we divide a MAC into two parts which can be verified individually, in case a link breaks. This allows us to mitigate possible jamming attacks or link failures on one of the two channels. In that case, however, only one half of the MAC is available which decreases the security level. By choosing an appropriate MAC length, the decrease of security can be mitigated. AUTOSAR states that MAC sizes of at least 64 bits provide sufficient protection against brute-force attacks by NIST [6]. In order to stick to these security guidelines, the size of the assembled MAC should be 128 bits, so that it can be split without raising security concerns.

5.3 Analysis of FlexRay's future

FlexRay's future is often viewed in a pessimistic way, because *Automotive Ethernet* is considered to be the next-generation communication standard in road vehicles. The envisioned future car does not any longer rely on current automotive protocols, such as LIN, CAN, MOST and FlexRay, but fully follows an Ethernet-driven approach. In fact, the in-vehicle communication is going to change massively in the next decades, but we doubt that current protocols will entirely disappear in the future, as they have been optimized for the vehicular use case with a special focus on safety for decades.

Ethernet is a mature and general-purpose protocol which has been used for many years in other domains. It is cheap and offers a variety of tailored protocols for different scenarios. Its adaptability makes it flexible for upcoming challenges, such as the rising data volumes that have to be transferred in real-time through the vehicle network. In recent years, automotive manufacturers have started using Ethernet in modern vehicles. Different shapes of Automotive Ethernet exist, e.g.BroadR-Reach [7]. A prominent application for Ethernet is the infotainment system and the transport of real-time sensor data.

In particular, we expect those manufacturers who use FlexRay in cars today (e.g. Mercedes, Audi, BMW, Porsche) to let FlexRay coexist with Ethernet due to a number of different reasons. First, the establishment of Ethernet in the vehicle industry is a slow and time-consuming process, especially when it comes to safety-critical applications. The supply chain is long and rather inflexible to sudden changes. Second, FlexRay has inherent advantages for the automotive use case, which makes it suitable for highly safety-critical and deterministic applications. For instance, it has guaranteed latencies.

We expect FlexRay to be deployed mainly for those applications which are responsible for driving dynamics. Another indicator for future deployment of FlexRay are current research projects (e.g. UNICARagil [21]) which incorporate FlexRay next to Ethernet.

6 CONCLUSION & FUTURE WORK

In this paper, we presented techniques to secure the automotive FlexRay protocol. As future cars will be intelligent and autonomous systems of systems and form part of the Internet of Vehicles, secure

in-vehicle communication is a fundamental feature. Otherwise, the vehicle's safety cannot be fully guaranteed, posing a severe threat to both the passengers and their environment. The co-engineering of security and safety is crucial to provide a safe road vehicle. We ensure this by using the dual-channel mode for concurrent transmissions of message authentication codes. In particular, we divide MACs into two truncated parts which are, then, individually transmitted in parallel over both channels. In case one channel is jammed or fails, data can still be authenticated using the other channel. Additionally, we suggest to associate cryptographic keys with time slots instead of using global keys, in order to scale independently of the number of network nodes. In case more nodes are added to the network, the amount of keys remains the same. Hash-based key chains allow fast key changes at runtime and simultaneously reduce the network traffic. In that way, additional security solutions do not collide with real-time driving operations.

In the future, we will explore how gateways, which translate between different automotive protocols, can be secured. Additionally, we aim to implement our techniques on real hardware.

ACKNOWLEDGMENTS

This work has been accomplished within the project "UNICARagil" (FKZ 16EMO0286). We acknowledge the financial support for the project by the Federal Ministry of Education and Research of Germany (BMBF).

REFERENCES

[1] 1991. CAN Specification. Robert Bosch GmbH. (1991). version 2.0.
[2] 2003. LIN Specification Package. LIN Consortium. (2003). revision 2.0.
[3] 2005. FlexRay Communications System: Protocol Specification. FlexRay Consortium. (2005). version 2.1 revision A.
[4] 2010. MOST Specification. MOST (Media Oriented Systems Transport) Cooperation. (2010). revision 3.0 E2.
[5] 2013. ISO Standard 17458–Road Vehicles–FlexRay Communications System. International Organization for Standardization (ISO). (2013).
[6] 2017. Specification of Secure Onboard Communication. AUTOSAR (AUTomotive Open System ARchitecture) Partnership. (2017). release 4.3.1.
[7] Broadcom Corporation. 2014. BroadR-Reach Physical Layer Transceiver Specification For Automotive Applications. (2014). http://www.ieee802.org/3/1TPCESG/public/BroadR_Reach_Automotive_Spec_V3.0.pdf version 3.0.
[8] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In *Proceedings of the 20th USENIX Conference on Security (SEC'11)*. USENIX Association, Berkeley, CA, USA. http://dl.acm.org/citation.cfm?id=2028067.2028073
[9] Danny Dolev and Andrew C. Yao. 1983. On the Security of Public Key Protocols. *IEEE Transactions on Information Theory* 29, 2 (March 1983), 198–208. https://doi.org/10.1109/TIT.1983.1056650
[10] Samir Fassak, Younes El Hajjaji El Idrissi, Noureddine Zahid, and Mohamed Jedra. 2017. A Secure Protocol for Session Keys Establishment Between ECUs in the CAN Bus. In *2017 International Conference on Wireless Networks and Mobile Communications (WINCOM)*. 1–6. https://doi.org/10.1109/WINCOM.2017.8238149
[11] Bogdan Groza, Stefan Murvay, Anthony van Herrewege, and Ingrid Verbauwhede. 2012. LiBrA-CAN: A Lightweight Broadcast Authentication Protocol for Controller Area Networks. In *Cryptology and Network Security*, Josef Pieprzyk, Ahmad-Reza Sadeghi, and Mark Manulis (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 185–200. https://doi.org/10.1007/978-3-642-35404-5_15
[12] Ahmed Hazem and HA Fahmy. 2012. LCAP - A Lightweight CAN Authentication Protocol for Securing In-Vehicle Networks. In *10th Embedded Security in Cars Conference (ESCAR)*.
[13] Charlie Miller and Chris Valasek. 2015. Remote Exploitation of an Unaltered Passenger Vehicle. Black Hat USA, Mandalay Bay, Las Vegas, NV, USA. (2015).
[14] Ahmed Refaat Mousa, Pakinam NourElDeen, Marianne Azer, and Mahmoud Allam. 2016. Lightweight Authentication Protocol Deployment over FlexRay. In *Proceedings of the 10th International Conference on Informatics and Systems (INFOS '16)*. ACM, New York, NY, USA, 233–239. https://doi.org/10.1145/2908446.2908485
[15] Sen Nie, Ling Liu, and Yuefeng Du. 2017. Free-Fall: Hacking Tesla from Wireless to CAN Bus. Black Hat USA, Mandalay Bay, Las Vegas, NV, USA. (2017).
[16] Stefan Nürnberger and Christian Rossow. 2016. – vatiCAN – Vetted, Authenticated CAN Bus. In *Cryptographic Hardware and Embedded Systems – CHES 2016*, Benedikt Gierlichs and Axel Y. Poschmann (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 106–124. https://doi.org/10.1007/978-3-662-53140-2_6
[17] Andreea-Ina Radu and Flavio D. Garcia. 2016. LeiA: A Lightweight Authentication Protocol for CAN. In *Computer Security – ESORICS 2016*, Ioannis Askoxylakis, Sotiris Ioannidis, Sokratis Katsikas, and Catherine Meadows (Eds.). Springer International Publishing, 283–300. https://doi.org/10.1007/978-3-319-45741-3_15
[18] Anthony Van Herrewege, Dave Singelee, and Ingrid Verbauwhede. 2011. CANAuth-a Simple, Backward Compatible Broadcast Authentication Protocol for CAN Bus. In *ECRYPT Workshop on Lightweight Cryptography*.
[19] Paula Vasile, Bogdan Groza, and Stefan Murvay. 2015. Performance Analysis of Broadcast Authentication Protocols on CAN-FD and FlexRay. In *Proceedings of the 2015 Workshop on Embedded Systems Security (WESS'15)*. ACM, New York, NY, USA, Article 7, 8 pages. https://doi.org/10.1145/2818362.2818369
[20] Qiyan Wang and Sanjay Sawhney. 2014. VeCure: A Practical Security Framework to Protect the CAN Bus of Vehicles. In *2014 International Conference on the Internet of Things (IOT)*. 13–18. https://doi.org/10.1109/IOT.2014.7030108
[21] Timo Woopen, Bastian Lampe, Torben Böddeker, Lutz Eckstein, Alexandru Kampmann, Bassam Alrifaee, Stefan Kowalewski, Dieter Moormann, Torben Stolte, Inga Jatzkowski, Markus Maurer, Mischa Möstl, Rolf Ernst, Stefan Ackermann, Christian Amersbach, Hermann Winner, Dominik Püllen, Stefan Katzenbeisser, Stefan Leinen, Matthias Becker, Christoph Stiller, Kai Furmans, Klaus Bengler, Frank Diermeyer, Markus Lienkamp, Dan Keilhoff, Hans-Christian Reuss, Michael Buchholz, Klaus Dietmayer, Henning Lategahn, Norbert Siepenkötter, Martin Elbs, Edgar v. Hinüber, Marius Dupuis, and Christian Hecker. 2018. UNICARagil - Disruptive Modular Architectures for Agile, Automated Vehicle Concepts. In *27th Aachen Colloquium Automobile and Engine Technology 2018*. Aachener Kolloquium Fahrzeug- und Motorentechnik GbR, Aachen, Germany, 663–694. https://doi.org/10.18154/RWTH-2018-229909