Security solutions for the CAN bus, bringing authentication to in-vehicle networks

Bogdan Groza and Pal-Stefan Murvay

Vehicles cannot be secured as long as their core, i.e., the in-vehicle network, remains insecure. The growing number of attacks reported each year show that, invariantly, in-vehicle buses are not isolated from the outside world. By exploiting their lack of security, adversaries can gain control over virtually any functionality inside the car. We discuss the most promising approaches for assuring security on the CAN-bus after a first decade of attacks and security proposals. Most of the proposals are based on cryptographic mechanisms, but this is not all as some exploit the physical layer or even physical characteristics of the controllers. The surveyed solutions prove a significant degree of maturity and sophistication which suggests that the moment for adoption and standardization by the industry should come.

Motivation, the CAN bus as attack surface

Security through isolation has always been an illusion. In the era of cyber-attacks, complex incidents such as the Stuxnet worm proved that even isolated facilities cannot stay secure from attacks orchestrated by strong-willed outsiders. Cars are no exception from this.

The first attack on in-vehicle networks that we could trace back was completing a rather mundane task: playing with the electric window lift [5]. Just a few years later, the first comprehensive analysis of in-vehicle security [7] demonstrates the corruption of various modules of a real-world car including safety critical components such as the engine control module, the brake control module and the body control module, etc. Currently, dozens of attacks are disclosed each year in research publications or through the news. A practical survey of in-vehicle vulnerabilities can be found in [10].

Needless to say, the vast majority of the attacks were launched through the in-vehicle network, e.g., the CAN bus (Controller Area Network), that mediates access to all of the existing modules and functionalities. Access to the in-vehicle network can be achieved through the diagnosis connection or by directly tapping the bus wires and, in case of remote connectivity, this can be done even from the outside. Once access to the bus is established, virtually all modules employing CAN-based communication are prone to attacks, the adversary being able to lock the brakes, steer the car, kill the engine, virtually controlling the car at his will. This view is suggested in Figure 1.

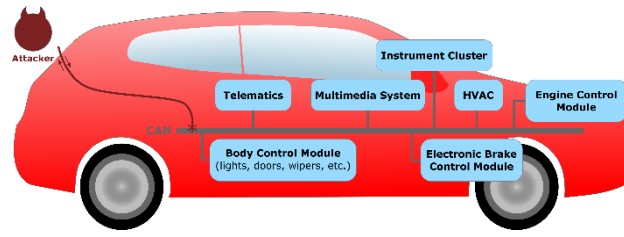

Figure 1 *Automotive modules attacked via the CAN bus*

The CAN bus is the workhorse behind most of the existing in-vehicle networks. It was designed by Bosch since 1983 and proved to be so successful that today it is present in every car on the market. Newer alternatives such as FlexRay or BroadR-Reach (an Ethernet based technology) bring more bandwidth, but at a higher cost without improving anything in terms of security. Thus, the large majority of attacks reported on the CAN bus are valid in all other existing in-vehicle network embodiments and, more important, the countermeasures proposed for CAN are largely extensible to all other in-vehicle buses.

The CAN bus is a two wire broadcast bus as suggested in Figure 2. At most 64 bits of data can be carried by one frame. Recently, CAN-FD (CAN with Flexible Data-Rate) was introduced as an alternative to CAN and it allows a higher data rate during the transmission of the data field which can be extended to 512 bits (64 bytes).

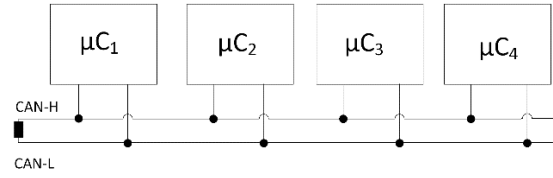

Figure 2 *Topology of the CAN bus*

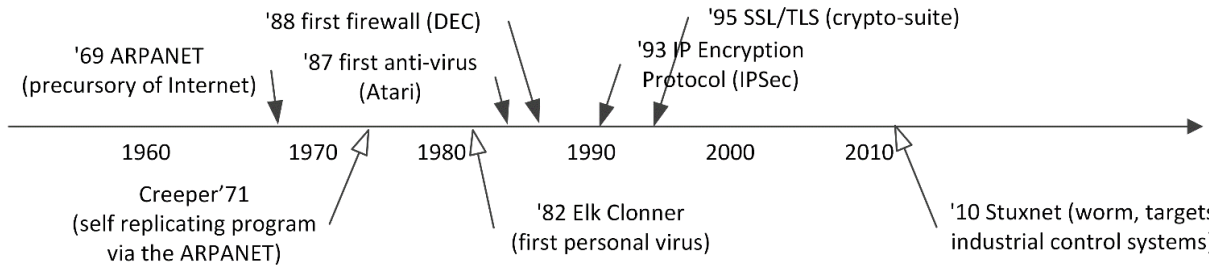

Figure 3 *Evolution of some security threats and defence mechanisms in the computer industry*

This makes CAN-FD more suitable for security at the application layer that incurs higher payloads for the messages. The CAN bus was designed with reliability in mind, for which a 15 bit CRC that accounts for transmission errors is simple and effective, but it has no intrinsic security mechanism.

A historical perspective

The over-increased connectivity is what opens doors to outsiders. In Figure 3 we depict several steps in the evolution of some security threats and defense mechanisms from the computer industry. From the development of the ARPANET (a foundational brick of today's Internet) it took a bit more than a decade for the first computer viruses to appear and the first defense mechanisms, i.e., firewalls and anti-viruses, to be put in place. In the 90's the need for cryptographic security become obvious and the development of the first crypto-suites started, e.g., IPSec and SSL/TLS. Today, we cannot imagine the Internet without these security mechanisms and security through isolation is not an option in the computer industry.

Cars will likely follow the same path. Especially in the era when remote functionalities for diagnosis, software updates, vehicle access, etc., become a must. If in-vehicle networks follow computer networks at one or two decades later, then we can expect the first in-vehicle malware to arrive in the next few years. This clearly leads to the need for devising security for in-vehicle buses, in particular for the CAN bus.

Some of the academic research efforts for bringing security mechanisms for the CAN bus are summarized in Figure 4 in a potential chronological order. We will discuss all these proposals in brief in what follows. Adoption by the industry requires standardization and the good news is that the AUTOSAR (AUTomotive Open System ARchitecture) standard includes specifications for cryptographic support since version 4.2.2 in 2015. Still, for the moment there is no standardized cryptographic protocol for in-vehicle buses.

In-vehicle ECUs are ready for cryptography

When it comes to performance, in-vehicle ECUs (Electronic Control Units) kept up with the increasingly demanding industry requirements. Coping with the demands of cryptographic algorithms comes within reach for more and more devices as operating frequencies and memory sizes are continuously growing.

To illustrate capabilities of automotive microcontrollers in handling cryptography we present in Table 1 the execution speed for several cryptographic primitives on four automotive-grade devices using an 8 byte input (i.e., the maximum payload of a standard CAN frames).

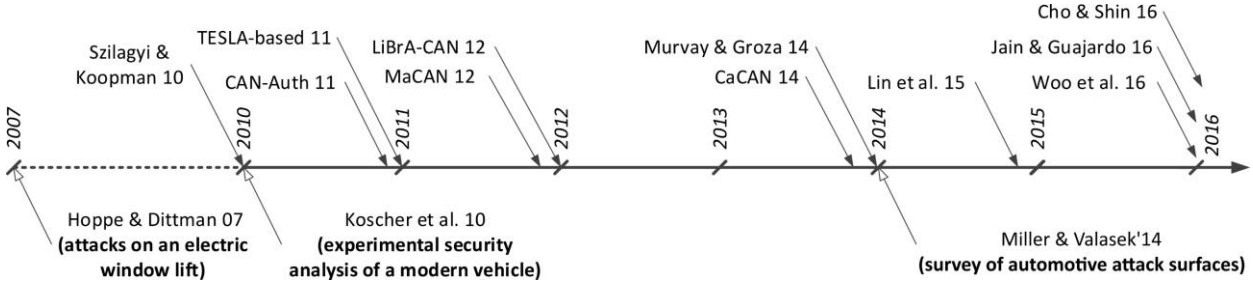

Figure 4 *Protocol proposals for assuring CAN-bus security in potential chronological order (by publication year), following the first reported attacks on in-vehicle networks*

Platform	SHA1	SHA256	SHA3-256	HMAC-SHA1	HMAC-SHA256	HMAC-SHA3	AES
S12XD	856µs	1.754ms	56ms	3.815ms	6.950ms	113.4ms	663µs
MPC5606B	958µs	604µs	29.7ms	3.935ms	2.350ms	63.1ms	934µs
TMS570LC	76µs	83.8µs	2.21ms	376µs	347µs	4.815ms	142µs
TC1797	48.6µs	57.7µs	5.23ms	213µs	223µs	10.62ms	55.7µs

Table 1 *Computational performance on automotive-grade controllers on various cryptographic primitives with 64 bit inputs*

Four representative cores are taken into account in Table 1 coming from both low-end and high end platforms: i) Freescale/NXP S12XD a member of the Freescale S12 family used in various powertrain, chassis and safety applications, ii) Freescale/NXP MPC5606B a general purpose automotive microcontroller, iii) Infineon TC1797 a powertrain, chassis and safety applications controller, iv) Texas Instruments TMS570LC457, a microcontroller designed for automotive safety-critical applications featuring a Cortex-R5 ARM core.

The results show that encryption algorithms such as the AES take a processing time in the order of dozens of micro-seconds on high-end platforms and hundreds for the low-end ones. Similar results are achievable for hash functions such as SHA1 and SHA256 or Message Authentication Codes (MACs) which proves that they are suitable for practical applications. The newer SHA3 standard has a somewhat poorer performance but is still within reach. We exclude SHA3 from the graphical depictions in Figure 5 to allow a clearer comparison between the remaining candidates. Additionally, in Figure 5 we add the computational time on 64 byte inputs, i.e., the size of a CAN-FD frame.

Cryptographic security at the application layer

Using cryptography at the application layer is the most natural choice and is in-line with what was already done in computer networks. Due to the limited bandwidth and also due to existing computational constraints on vehicular ECUs, standard MACs are the cryptographic function of choice for assuring message authenticity.

MACs require a secretly shared key between the participants. The MAC function is applied over the message and the secret key to generate an authentication tag that is verified by re-computing it based on the received message and the known secret key. The HMAC is the most popular choice for a MAC. It requires applying a cryptographic hash function twice along with a secret key *k* and two constant padding values *ipad* and *opad* on the message:

$$HMAC(k, message) = Hash\big(k \oplus ipad || Hash(k \oplus opad || message)\big)$$

While most of the protocol descriptions that follow employ regular MACs, the difference between them is in the way the secret keys are shared and/or used for computing the MACs.

Regular key sharing and MACs

A unique secret key for the entire CAN network is not a good alternative since if a single node is corrupted, security is lost for the entire bus. The most obvious procedure is sharing secret keys pair-wisely between nodes. This mechanism is employed by the following schemes and suggested in Figure 6 (left) where each ECU holds one secret key shared with each of the other ECUs.

Voting schemes for time-triggered communication were introduced by Szilagyi and Koopman in [13]. The scheme is intended for the generic time-triggered communication present

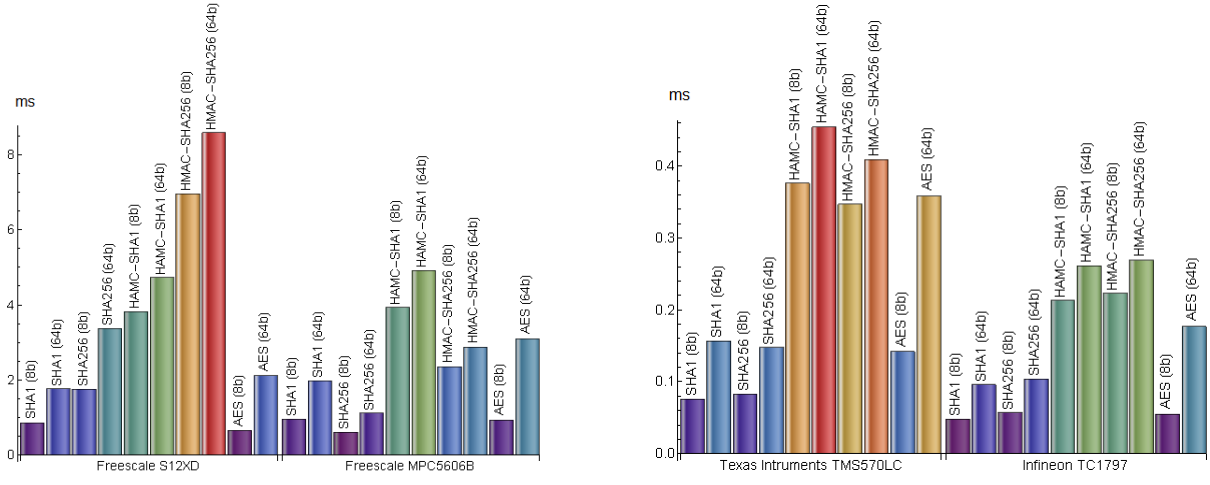

Figure 5 *Computational time (in milliseconds) for low-end cores (left) and high-end cores (right)*

in TT-CAN or FlexRay, etc. Due to the limited space, the tags are truncated, e.g., 3 MACs each of 8 bits are fitted at the end of a single frame in case of 3 receivers. Since each frame carries only a small amount of authentication information, a message needs to accumulate a sufficient number of votes (i.e., authentication tags) to be deemed authentic. The idea of voting does not really seem suited for the nature of CAN as the real-time nature of communication doesn't allow enough time for nodes to cast votes. Voting for past received messages is also suggested [13] but nodes may not share the same receive history due to ID filtering at the hardware level or because nodes may go into bus off. This proposal may be limited in application to the CAN bus, but it is the first research effort.

MaCAN is proposed in [4]. The protocol employs shared keys between nodes and MACs derived from block ciphers, i.e., the CMAC construction (this saves some of the computational time as block ciphers such as the AES can be used). To cope with the limited size of the data-field MACs are truncated to 4 bytes. The authors of MaCAN [4] also suggest that nodes can be grouped under the same key if they share the same trust level, but no practical insights are given on how to decide the trust level. MaCAN uses the straight-forward way of sharing keys between nodes and fitting a truncated MAC in half of the data-field.

A more recent MAC-based scheme accompanied by experimental results for the newer CAN-FD can be found in [15].

Group key sharing

The main limitation of pairwise key sharing is that the limited space of a single tag is equally split between the receivers. For example, if the MAC space is limited to 24 bits and there are 3 tags/receivers, each will get only 8 bits of authentication. This is clearly too low. But more can be done than the simple pair-wise sharing of keys between nodes and more efficiency can be gained from a single authentication tag. The simple contrast between pair-wise key sharing and group key-sharing in Figure 6 shows why this is the case. In Figure 6 (right) each ECU groups the other ECUs into groups of size 3. Key k_{11} is shared by ECU_1 with ECU_2 and ECU_3, key k_{12} with ECU_2 and ECU_4, etc. When sending a MAC with the 3 keys, i.e., k_{11}, k_{12}, k_{13}, each of the 3 other ECUs will be in possession of 2 keys out of the 3, thus gaining 16 bits of the tag rather than 8. The security level is thus doubled. In case of a single corrupted node, the security drops to 8 bits and if 2 corrupted nodes exist the security drops to 0. Since in-vehicle networks are built by reputable manufacturers, corrupted nodes must be in minority. Such a scheme has good security advantages in case of corrupted minorities. This is exactly the principle behind the next proposal.

LiBrA-CAN [2] is the first to propose a group key allocation procedure which mixes keys between groups of nodes (rather than sharing them pair-wisely). Besides mixing the keys between groups of nodes, LiBrA-CAN also makes use of a more advanced MAC construction which mixes the authentication tags allowing forgeries to be detected even if these are done for another key of the same mixed MAC. LiBrA-CAN is a more demanding security protocol for the CAN bus, less straight-forward to implement but with certain security advantages in front of the other solutions.

Time synchronization

The **TESLA** protocol [12] was a breakthrough in sensor networks. This protocol allows symmetric keys to be used for broadcast authentication by releasing them in a time-dependent fashion. It enables simple, cost-efficient symmetric cryptographic functions to be used without secret keys for assuring broadcast authentication to multiple receivers. Bringing this protocol to the CAN bus was considered in [3] where several trade-offs are studied. One drawback in adopting this protocol for in-vehicle networks is that it achieves authentication with a small delay as keys are released at fixed time intervals. This delay would usually be in the order of 1–10 *ms* which may be small enough to preserve the real-time nature of CAN. The results in [3] prove that such protocols are within reach for in-vehicle networks.

Efficient signal allocation

Clearly, security mechanisms add overheads that can impede system performance. Lin et al. [9] discuss how to deal with both security and safety constraints. The authors consider tasks on each ECU as source or destination for signals and a path as being an interleaving sequence of tasks and signals. Path-based security constraints are then formulated and a heuristic

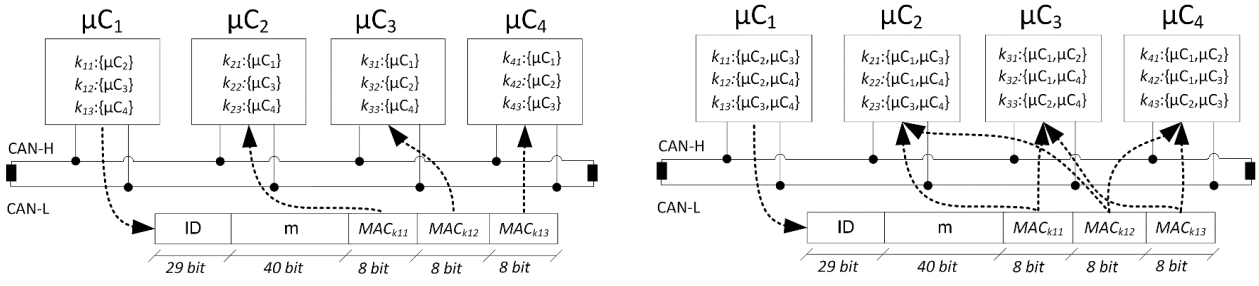

Figure 6 *Pair-wise key sharing between nodes (left) vs. group key sharing as proposed in LiBrA-CAN (right)*

algorithm is proposed to efficiently find a solution to this problem. While this line of work does not come with a new security protocol (it relies on regular MACs and shared keys), the constraint based signal allocation can be applied to any of the previous mechanisms for increasing their efficiency.

Security at the physical layer

Sharing the keys

How to share keys between ECUs is a relevant issue. Standard cryptographic mechanism, e.g., public-key mechanism, can be used for this but come with high overheads. Recently, in [6] a novel mechanism is explored with keys exchanged in a secure manner by exploiting the physical properties of the signal on the CAN bus. Briefly, since dominant bits overwrite recessive bits, two nodes that send a message at the same time can still ascertain part of what the other has sent. This is further explored by [6] with a tree-based key-sharing mechanism where the leaf nodes of the tree are the physical nodes while all the other virtual nodes in the tree correspond to logical entities that can be emulated by any physical node connected to it. The results presented in [6] evaluate the cost for *M* nodes to share an *n* bit key, showing certain advantages for the tree-based scheme.

Obstructing forged frames at the physical layer

CaCAN [8] introduces a centralized view over the authentication process. In this protocol a central node verifies the authentication tags of each frame and if authentication fails, the frames are discarded with error flags. This procedure has the merit of requiring a single monitor node with higher computational power for this purpose. However, an adversary that removes this node from the bus can take full control of the network since there is no way for the other nodes to decide if a frame is authentic or not. This may be a serious limitation for practical deployments and likely the only one that may stop CaCAN from becoming an industry-standard solution.

Hiding authentication bits with CAN+: CAN-Auth

CANAuth proposes the use of an ID-oriented key allocation procedure [14] and a non-standard CAN derivative called CAN+. From a cryptographic perspective CANAuth relies on simple message authentication codes and shared keys, we place it here since the cryptographic constructions behind it bring nothing new while the idea of using a new physical layer is what makes it of significant interest. CAN+ allows additional bits to be inserted during the time of a single bit. Concretely, as the sample point for the CAN bus is at around 75% of the bit-time, the CAN+ transmission window is placed at between 15%–55% of the nominal bit time. This allows for authentication bits to be send in a stealthy manner making the solution back-ward compatible. Convincing experiments are provided on an FPGA implementation in [14]. The main limitation of this solution is the uncertainty regarding the practical adoption of CAN+. With CAN-FD already released, it seems unlikely for CAN+ to become a successor of CAN.

Security by physical characteristics

Physical characteristics of electronic devices open a new vista for security applications. In the recent years, physically unclonable functions (PUF) gained a significant momentum in the security industry. We discuss now the only two lines of work that bring CAN security down to physical characteristics.

Fingerprinting physical signals

The use of physical signal characteristics on CAN to distinguish between sender nodes is discussed for the first time in [11]. The main advantage of this procedure is that it does not require cryptography, thus it removes the problem of sharing cryptographic keys or adding bus overheads and is fully backward compatible. The research in [11] shows that one can successfully distinguish between CAN nodes with good success probability by applying standard mathematical tools on the electrical signal characteristics, e.g., mean-square errors (MSE) or convolutions. Frames that do not match the expected signal characteristics can be destroyed or signaled with error flags. Figure 7 graphically depicts this separation for transceivers from an USB-to-CAN device (left) and Freescale S12 development boards (right) based on mean square errors as shown in [11]. The distance between each frame F_β and the fingerprint Sig_α is computed over each sample of the signal as:

$$MSE(Sig_\alpha, F_\beta) = \sum_{i=1}^{\ell} (Sig_\alpha[i] - F_\beta[i])^2$$

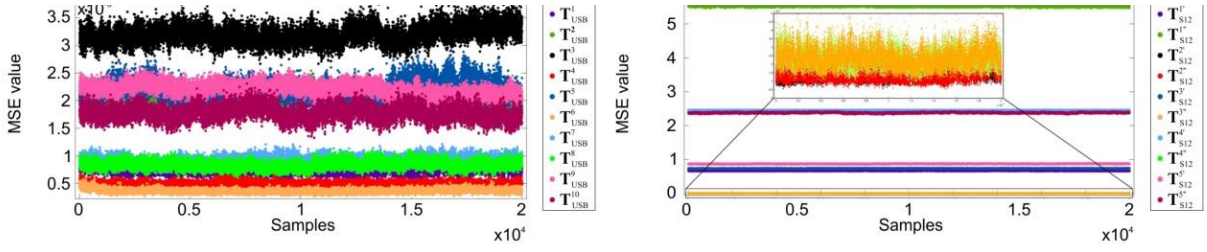

Figure 7: MSE based separation with an USB-to-CAN device (left) and a Freescale S12 development board for 2×104 recorded values (as depicted in [11])

Experimental results are presented in [11] for two CAN transceivers PCA82C251 (a high-speed CAN transceiver from USB-to-CAN devices) and TJA1054T (a low-speed CAN transceiver from Freescale S12 development boards). With the exception of 3 transceivers from a set of 10, identification rates are over 90%. Even with the colliding fingerprints of the 3 transceivers, by changing the ID of message the overlap rate drops to 0%. This solution may not be easy to implement but it is the first not to rely on cryptography for assuring security on the CAN bus.

Using clock skews to detect intrusions

Using clock skews is the most recent proposal for detecting intrusions on the CAN bus [1]. This mechanism was successfully explored in the past for source identification in computer networks and mobile phones, but never exploited for in-vehicle networks. The main idea is that there are no physically identical oscillators and variations in the order of several parts-per-million (ppm) can be used to separate between devices. This is exploited in [1] by noticing the cyclic nature of the communication on the CAN bus. That is, periodic messages are sent at exact time intervals, e.g., δ, 2δ, 3δ, etc. However, the sender ECU, rather than sending the message from the i^{th} cycle at time $i\delta$ will send it at $i\delta + O_i$ where O_i is his clock offset. Experimental data from [1] shows that clock drifts are separable in the order of dozens of ppm on several real-world vehicles. For example in the Honda Accord (2013) four sources are determined at: 78.4ppm, 199.8ppm, 265.7ppm and 95.78ppm. Further difficulties appeared in the Toyota Camry (2010) where two clock skews differed by less than 3%, namely 345.ppm vs. 334.1ppm, which would make these ECUs hard to distinguish. Given the high number of existing ECUs, it seems reasonable to assume that such collisions in the clock drifts may be usual in practice. But the method seems very promising for future investigations.

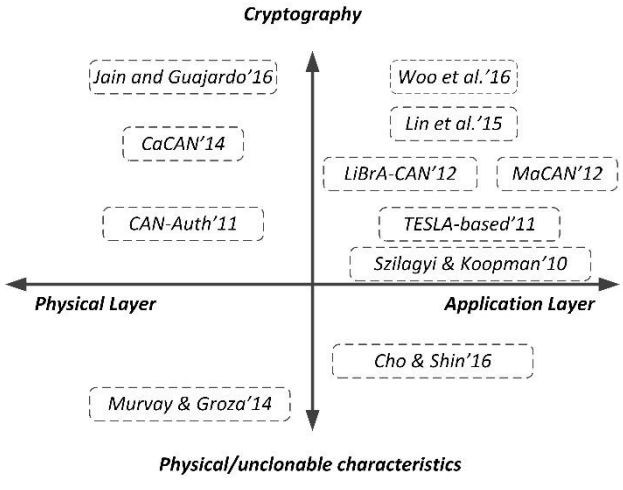

Figure 8 *Security proposals for CAN, separated by direction toward the physical layer vs. the application layer and cryptographic techniques vs. physical/un-clonable characteristics*

Discussion and conclusion

Assuming a similar evolution to that of computer networks, cryptography will become mandatory on in-vehicle buses. The inclusion of cryptographic interfaces in the AUTOSAR standard is a first foundational brick. CAN-FD, the newer embodiment of CAN, offers plenty of space in the frame for adding modern security mechanisms and there are already many research proposals to be considered for industry adoption.

Figure 8 rounds up the discussed solutions and separates them along the following lines: application layer vs. the physical layer and by the use of cryptography vs. physical characteristics. There are no doubts that cryptographic MACs will stay behind assuring authentication, the challenge remains in how they are used, i.e., the concrete authentication protocol. In this respect, group key sharing offers more advantages in front of the basic pairwise key sharing. Given the cyclic nature of communication on in-vehicle buses, TESLA-like protocols should gain more momentum and be of particular interest for time-triggered networks, e.g., FlexRay. Physical properties of the signal or of existing electronics, e.g., clock drifts, can form the basis of intrusion detection mechanisms.

References

[1] K.-T. Cho and K. G. Shin. Fingerprinting electronic control units for vehicle intrusion detection. In 25th USENIX Security Symposium, 2016.

[2] B. Groza, P.-S. Murvay, A. Van Herrewege, and I. Verbauwhede. LiBrA-CAN: a lightweight broadcast authentication protocol for controller area networks. In 11th International Conference on Cryptology and Network Security, CANS 2012, Springer-Verlag, LNCS, 2012.

[3] B. Groza and S. Murvay. Efficient protocols for secure broadcast in controller area networks. IEEE Transactions on Industrial Informatics, 9(4):2034–2042, 2013.

[4] O. Hartkopp, C. Reuber, and R. Schilling. MaCAN-message authenticated CAN. In 10th Int. Conf. on Embedded Security in Cars (ESCAR 2012), 2012.

[5] T. Hoppe and J. Dittman. Sniffing/replay attacks on can buses: A simulated attack on the electric window lift classified using an adapted cert taxonomy. In Proceedings of the 2nd workshop on embedded systems security (WESS), pages 1–6, 2007.

[6] S. Jain and J. Guajardo. Physical layer group key agreement for automotive controller area networks. In Conference on Cryptographic Hardware and Embedded Systems, 2016.

[7] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, et al. Experimental security analysis of a modern automobile. In Security and Privacy (SP), 2010 IEEE Symposium on, pages 447–462. IEEE, 2010.

[8] R. Kurachi, Y. Matsubara, H. Takada, N. Adachi, Y. Miyashita, and S. Horihata. CaCAN - centralized authentication system in CAN (Controller Area Network). In 14th Int. Conf. on Embedded Security in Cars (ESCAR 2014), 2014.

[9] C.-W. Lin, Q. Zhu, and A. Sangiovanni-Vincentelli. Security-aware modeling and efficient mapping for CAN-based real-time distributed automotive systems. IEEE Embedded Systems Letters, 7(1):11–14, 2015.

[10] C. Miller and C. Valasek. A survey of remote automotive attack surfaces. Black Hat USA, 2014.

[11] P.-S. Murvay and B. Groza. Source identification using signal characteristics in controller area networks. IEEE Signal Process. Letters, 21(4):395–399, 2014.
[12] A. Perrig, R. Canetti, J. Tygar, and D. X. Song. Efficient authentication and signing of multicast streams over lossy channels. In IEEE Symposium on Security and Privacy, pages 56–73, 2000.
[13] C. Szilagyi and P. Koopman. Low cost multicast authentication via validity voting in time-triggered embedded control networks. In Proceedings of the 5th Workshop on Embedded Systems Security, page 10. ACM, 2010.
[14] A. Van Herrewege, D. Singelee, and I. Verbauwhede. CANAuth-a simple, backward compatible broadcast authentication protocol for CAN bus. In 9-th Embedded Security in Cars Conference, 2011.
[15] S. Woo, H. J. Jo, I. S. Kim, and D. H. Lee. A practical security architecture for in-vehicle CAN-FD. IEEE Transactions on Intelligent Transportation Systems, 17(8):2248–2261, Aug 2016.

Acknowledgements

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS-UEFISCDI, project number PN-II-RU-TE-2014-4-1501 (2015-2017).

Bogdan Groza *(bogdan.groza@aut.upt.ro) is an associate professor at Politehnica University of Timisoara (UPT). He received his Dipl.Ing. and Ph.D. degree from UPT in 2004 and 2008 respectively. In 2016 he successfully defended his habilitation thesis having as core subject the design of cryptographic security for automotive embedded devices and networks. He has been actively involved inside UPT with the development of laboratories by Continental Automotive and Vector Informatik, two world-class manufacturers of automotive software. He currently leads the CSEAMAN project, a 2 years research program (2015-2017) in the area of automotive security.*

Pal-Stefan Murvay *(pal-stefan.murvay@aut.upt.ro) is an assistant professor at Politehnica University of Timisoara (UPT). He graduated his B.Sc and M.Sc studies in 2008 and 2010 respectively and received his Ph.D. degree in 2014, all from UPT. He has a 9-year background as a software developer in the automotive industry as former employee of Continental Corporation (2005- 2014). His current research interests are in the area of automotive security and works as a postdoctoral researcher in the CSEAMAN project.*