CRYPTOGRAPHIC SECURITY FOR AUTOMOTIVE SYSTEMS

HABILITATION THESIS

Bogdan Groza

Politehnica University of Timişoara
Faculty of Automatics and Computers
bogdan.groza@aut.upt.ro

December, 2015

Summary

The thesis addresses our research between 2010-2015 at Politehnica University of Timisoara, focusing on the design of cryptographic protocols for assuring security on in-vehicle buses (e.g., the CAN bus) and various automotive components or functionalities (e.g., tire pressure monitoring sensors, vehicle access control by smart-phones). This constitutes only a part of our research which targeted several directions ranging from theoretical cryptography and formal methods up to more practical subjects such as network and mobile systems security (these are briefly accounted in an overview section).

In the recent years, it has become increasingly obvious that vehicle evolution brings many similarities to that of modern computers. Not more than a century ago, computers were mere mechanical machines, then they turned into complex electronics and today they are loaded with complex software that (arguably) surpasses the complexity of the electronics behind it. This of course does not diminish the importance of the hardware without which they cannot function, but opens an entirely new vista for applications that have tremendously improved the quality of our life. Similarly, in the past decades, cars turned from mechanical devices into complex electronic devices and now they are loaded with hundreds of functionalities that are implemented in the software. These functionalities reside on dozens (even hundreds) of miniature devices, called Electronic Control Units (ECUs), that are spread inside the car and connected via a complex internal network. To make things even more interesting from a security perspective, part of this network is exposed to outsiders (i.e., potential adversaries) via wired channels (e.g., OBD ports) or via wireless interfaces (e.g., 3G, Bluetooth). The number of reported attacks has drastically ascended in the past years, with recent reports showing how one can lock the engine, steering wheels or the brakes, listen to passengers conversation, etc., even from hundred of miles away. The dull security landscape of cars from the past, dominated by small frauds (mileage modification, car theft, etc.), turned interesting in the recent years once it become clear that adversaries can takeover a car and use it at will.

Surprisingly, security mechanisms are completely absent from vehicular buses, starting from traditional ones such as the CAN bus (Controller Area Network) up to the most recent developments such as CAN-FD or FlexRay. This is mostly due to several technical challenges: low bandwidth and processing power, low cost margins, slow standardization, etc. Our work is focused on the design of efficient broadcast authentication protocols taking into account the three most promising techniques: TESLA-like protocols based on key chains and time synchronization, group keying protocols where keys are shared between groups of nodes and one-time signature (an alternative which is quickly discarded). While TESLA-like protocols proved highly efficient in sensor networks, this does not seem to be the case for in-vehicle networks as authentication delays need to be kept small and this raises synchronization problems, if we increase the delays we hit memory issues as large amounts of data need to be buffered. Moreover, the busload is also increased by the release of the authentication keys. The most promising solution appears to be group keying, i.e., LiBrA-CAN. This protocol is based entirely on simple symmetric primitives and takes advantage

of two interesting procedures which we call key splitting and MAC mixing. Rather than achieving authentication independently on each node, we share keys between groups of nodes which leads to a higher security level in case of compromised nodes forming only a minority. Based on practical arguments, we recognize this assumption to be realistic for automotive networks. Subsequently, amalgamating regular message authentication codes with systems of linear equations increases the chances for a forgery to be detected. We present several protocol variants that are extremely flexible and set way for different trade-offs on bus load, computational cost and security level, taking into account the most recent developments such as the recently released CAN-FD standard. To asses the efficiency of the proposed solution the proposed protocols were tested on automotive-grade microcontrollers as well as via simulation with industry standard tools. By the use of the CANoe tool we were able to simulate bandwidth allocation for the proposed protocols on state-of-art buses such as CAN-FD and FlexRay. The practical results proved our intuitions from the synthetic comparison of the protocols, i.e., group keying (LiBrA-CAN) is the preferred protocol design.

Departing from in-vehicle buses, there are so may other automotive sub-systems that are still deprived of security functionalities, e.g., wireless sensors inside wheels, car keys, etc. Moreover, even when components have certain security features, the security threats are far from being removed, e.g., car theft and millage modifications are still common issues. Clearly, a system cannot be more secure than its weakest link and we need to design security for these components as well. Here our results are dispersed and address several subsystems starting from the generation of random numbers on embedded devices, smart-phone based vehicle access and security for wireless sensors. We do present our most recent contributions in the security of wireless communication interfaces used in Tire Pressure Monitoring Systems (TPMS). Our work starts from designing an efficient authentication protocol based on lightweight cryptographic designs and block cipher based message authentication codes. The experimental results show that the proposed solution can be handled by real world sensors and is more efficient than related proposals. The works on smart-phone based car access and on randomness for automotive grade controllers, are recent developments and joint works with the industry.

Sumar (limba română)

Teza curentă adresează eforturile noastre de cercetare din perioada 2010-2015 la Universitatea Politehnica Timişoara fiind focalizată pe construcţia de protocoale criptografice pentru securitatea reţelelor din vehicule (e.g., CAN bus) şi a diverselor componente automotive sau funcţionalităţi (e.g., senzori de monitorizare a presiunii în roţi, accesul la vehicule folosind telefoane inteligente). Aceasta constituie doar o parte a cercetărilor noastre în criptografie şi securitatea sistemelor din această perioadă, cercetare care a vizat diverse direcţii de la criptografie teoretică şi metode formale până la zone aplicative precum securitatea reţelelor şi a dispozitivelor mobile (acestea sunt pe scurt prezentate într-o secţiune dedicată unei priviri de ansamblu asupra rezultatelor autorului).

În anii recenţi a devenit din ce în ce mai evident că evoluţia vehiculelor se apropie de cea a calculatoarelor. Cu un secol în urmă, calculatoarele erau dispozitive pur mecanice, apoi ele au devenit dispozitive electronice complexe pentru ca astăzi să fie încărcate cu componente software a căror complexitate (posibil) depăşeşte complexitatea electronicii din fundal. Aceasta în nici un caz nu diminuează importanţa hardware-ului fără de care acestea nu ar putea funcţiona, dar deschide o nouă perspectivă pentru aplicaţii care au îmbunătăţit fabulos calitatea vieţii noastre. Similar, în deceniile anterioare, maşinile au devenit din dispozitive pur mecanice, dispozitive electronice complexe şi sunt încărcate cu sute de funcţionalităţi care rezidă pe duzini (chiar sute) de dispozitive miniaturale, numite ECU (Electronic Control Units), care sunt răspândite prin maşini şi conectate prin reţele interne complexe. Pentru a face lucrurile şi mai interesante din perspectiva securităţii, o parte din aceste reţele sunt expuse în exterior (i.e., către posibili adversari) prin canale cablate (e.g., port OBD) sau canale wireless (e.g., 3G, Bluetooth). Numărul de atacuri raportate a crescut drastic în anii trecuţi, lucrări recente arată cum atacatori pot bloca frânele, roţile sau motorul, asculta conversaţiile pasagerilor, etc., chiar şi de la sute de kilometrii distanţă. Peisajul sterp al securităţii maşinilor din trecut, dominat de infracţiuni minore (modificare kilometraj, furt de maşini, etc.) a început să devină fertil în anii recenţi, de îndată ce a devenit clar că adversarii pot prelua controlul unei maşini şi să o folosească după propria voie - toate acestea prin canale elctronice şi chiar de la distanţă.

Surprinzător, mecanismele de securitate sunt complet absente din magistralele de comunicaţii din vehicule, începând de la cele tradiţionale precum CAN (Controller Area Network) până la cele mai recente dezvoltări precum CAN-FD sau FlexRay. Aceasta se datorează multor provocări tehnice: lăţime de bandă scăzută, putere de procesare scăzută, margini de cost, standardizare lentă, etc. Lucrările noastre sunt focalizate pe construcţia unor protocoale de autentificare broadcast luînd în calcul cele mai promiţătoare trei tehnici: protocoale tip TESLA bazate pe lanţuri one-way şi sincronizare temporală, protocoale bazate pe distribuţia cheilor în sub-grupuri şi semnături one-time. În timp ce protocoalele de tip TESLA s-au dovedit a fi extrem de eficiente în reţele de senzori, acestea nu par să fie o alternativă bună în automotive: întârzierile de autentificare trebuie păstrate cât mai mici şi aceasta duce la probleme de sincronizare, dacă creştem aceste întârzieri, ajungem la limitări de memorie deoarece cantităţi mari de date trebuie păstrate într-un buffer. Mai mult,

încărcarea cu date a magistralei este sporită de eliberarea cheilor de sesiune. Cea mai promiţătoare metodă pare a fi gruparea cheilor pe sub-grupuri, i.e., LiBrA-CAN. Acest protocol este bazat în intregime pe primitive simetrice şi foloseşte două proceduri inovatoare: distributirea cheilor şi amestecarea MAC-urilor. În loc să axăm autentificarea independent pe fiecare nod (ce ar duce la un număr prea mare de chei) vom partaja cheile între grupuri de noduri ceea ce duce la un nivel de securitate mai ridicat în cazul în care nodurile corupte sunt în minoritate. Folosind argumente practice, această presupunere este demonstrabil corectă pentru reţele automotive. Mai departe, codurile standard de autentificare (MAC) sunt amalgamente folosind sisteme de ecuaţii liniare pentru a creşte şansele ca un fals să fie detectat. Prezentăm câteva variante de protocol care sunt flexibile şi deschid posibilitatea unor trade-off-uri între rata de date, încărcarea computaţională şi nivelul de securitate, luînd în calcul cele mai recente magistrale precum standardul CAN-FD sau FlexRay. Pentru a analiza eficienţa protocoalelor propuse acestea au fost testate pe microcontrollere de clasă automotive precum şi prin simulări folosind instrumente standard de simulare folosite în industrie. Prin folosirea CANoe a fost simulată utilizarea ratei de date pe magistrale state-of-the-art precum CAN-FD şi FlexRay. Rezultatele practice arată că intuiţia din comparaţia sintetică este corectă şi că alocarea cheilor pe grupuri este designul preferat de protocol.

Distanţându-ne de reţelele in-vehicle sunt atâtea alte subsisteme din automotive care sunt încă extrem de limitate în ceea ce priveşte funcţionalităţile de securitate, e.g., senzorii din roţi, chei wireless, etc. Mai mult, chiar şi în cazul componentelor care au funcţionalităţi de securitate, ameninţările sunt departe de a fi eliminate, e.g., falsificarea cheilor şi modificarea kilometrajului sunt încă probleme comune. În mod cert, un sistem nu poate fi mai sigur decât cea mai slabă verigă a sa şi trebuie să avem în vedere şi aceste componente. Aici rezultatele noastre sunt distribuite şi adresează câteva subiecte cum ar fi generarea numerelor aleatoare pe microcontrollere de clasă automotive, accesul la maşină folosind telefonul mobil şi securitatea senzorilor wireless. Vom prezenta contribuţiile noastre cele mai recente în securitatea intefeţelor wireless pentru senzori Tire Pressure Monitoring Systems (TPMS). Lucrările noastre au la bază construcţia unor protocoale de autentificare eficiente bazate pe designuri criptografice light-weight şi coduri de autentificare bazate pe coduri bloc simetrice, e.g., CBC-MAC. Rezultatele experimentale demonstrează că soluţiile propuse pot fi integrate în senzori din lumea reală si sunt mai eficiente decât cele propuse în alte lucrări. Lucrările legate de folosirea telefonului mobil pentru accesul maşinii şi generarea de numere aleatoare sunt lucrări rezultate din cooperarea recentă cu industria.

Acknowledgements

There are many names that I can acknowledge for collaborating with me through the years. Still, I feel that the list here should be limited to the ones that had a direct (scientific) impact on what is written in the thesis, this is a much shorter list.

Through the years, Marius Minea was a friend on which I could always rely for various research matters. While none of our several joint works (2009–2013) is a direct part of this thesis, our results on modelling protocol security are of relevance to the subject addressed here (I'll slightly touch several titles). In 2008 I started to work on embedded systems security with Ştefan Murvay, who was my student at that time and soon we become friends (sharing a lot of fine nature explorations together). Chapter 3 dedicated to in-vehicle network security is based on papers that are joint work with Ştefan. At the same time, I had the opportunity to work with Marius Cristea (who was pursuing his PhD) on network security, one of our works on mobile devices security is slightly touched in the thesis. Since 2012, my cooperation started to spread to various students: Cristina Solomon worked with me on TPMS security, Paula Vasile with simulations for in-vehicle protocols, Ciprian Gurban on remote vehicle diagnosis and George Tipa on randomness for automotive grade controllers.

My cooperation with foreign universities also needs to be acknowledged. Bogdan Warinschi hosted my visit to Bristol in 2011. In 2012, I visited René Mayrhofer in Linz at the University of Applied Sciences Upper Austria. Anthony van Herrewege and Ingrid Verbauwhede hosted my visit to KU Leuven in 2013. The joint works with them have contributed to my education as a researcher.

I am grateful to all of the above for giving me the opportunity to work with them!

Contents

Part I

Scientific results

Chapter 1

Overview of author's research results

This chapter enumerates the main results of the author from 2010 to 2015. For this period, 20 research papers (A1-A20) and 10 independent research presentations (P1-P10) cover the work described in this thesis. Part of them focus on the central body of work on automotive security (A1-A9 and P1-P5) and will be also detailed in the forthcoming sections. The second body of publications (A10-A20) and presentations (P6-P10) cover other lines of research in information security and cryptography, e.g., theoretical cryptography, formal methods, network and mobile devices security, etc., and are only briefly outlined in this overview section.

1.1 Research results in automotive security

RESEARCH PAPERS AND PATENT APPLICATIONS. Our work in automotive security started with the design of security protocols for in-vehicle networks, e.g., CAN. As a result of numerous discussions with the industry, including students and graduates employed in automotive companies, we have expanded our work to other areas with potential for relevant scientific contributions: security for wireless in-vehicle sensors, wireless automotive keys, remote vehicle diagnosis and randomness on embedded devices. We enumerate here the scientific contributions and point out the core ideas in each direction. Figure 1.1 graphically depicts our contributions around a schematic car body.

- **Designing authentication protocols for in-vehicle networks**, e.g., CAN (Controller Area Network) has been a main research topic since 2010. We have designed several protocols, starting from TESLA-like protocols to others with group keying and one-time signatures. My contributions in this line of work consist in the protocol designs and synthetic evaluation for protocol performances. A significant outcome of this line of work has been the PhD thesis of Stefan Murvay who has demonstrated the applicability of these ideas, achieving a practical implementation. Our most relevant papers are the ones below.

  [A1] Bogdan Groza, Stefan Murvay, *Broadcast Authentication in a Low Speed Controller Area Network*. Revised selected papers from SECRYPT'11, Springer CCIS Series, 2012.

  [A2] Bogdan Groza, Stefan Murvay, *Secure Broadcast with One-time Sigantures in Controller Area Networks*. Proceedings of International Conference on Availability, Reliability and Security (ARES'11), IEEE Comp. Soc., 2011.

[A3] Bogdan Groza, Stefan Murvay, *Efficient protocols for secure broadcast in Controller Area Networks*, IEEE Transactions on Industrial Informatics, Volume: 9, Issue: 4, Pages: 2034 - 2042, IEEE, 2013.

[A4] Bogdan Groza, Stefan Murvay, Anthony van Herrewege, Ingrid Verbauwhede, *LiBrA-CAN: a Lightweight Broadcast Authentication protocol for Controller Area Networks*, Proc. 11th Intl. Conf. on Cryptology and Network Security (CANS'12), Springer-Verlag, LNCS vol. 7712, 2012.

[A5] Stefan Murvay, Bogdan Groza; *Source Identification Using Signal Characteristics in Controller Area Networks*, Signal Processing Letters, Volume: 21, Issue: 4, Pages: 395 - 399, IEEE, 2014.

- **Simulation of authentication protocols for state-of-the-art in-vehicle networks, e.g., CAN-FD and FlexRay** with industry standard tools such as CANoe is fundamental. Previous to our recent work [95], none of the proposed protocols has been tested by such simulation tools. The main reason is that such simulation tools and the architecture of in-vehicle networks are usually out of reach for university researchers. This topic emerged in our research in late 2014, mainly due to the support received from Vector GmbH. The results below have been recently published.

  [A6] Paula Vasile, Bogdan Groza, and Stefan Murvay. *Performance analysis of broadcast authentication protocols on CAN-FD and FlexRay*, WESS: 10th Workshop on Embedded Systems Security (affiliated to ESWEEK 2015), 2015.

- **Designing security for vehicular wireless Tire Pressure Monitoring Systems (TPMS)** is a topic we have been following since 2010 when the first attacks were reported. After several attempts to secure industry cooperation and obtaining the needed experimental support, in 2014 we started work on a real-world implementation for security mechanisms. The results below have been recently published.

  [A7] Cristina Solomon, Bogdan Groza, *LiMon - lightweight authentication for tire pressure monitoring sensors*, 1st Workshop on the Security of Cyber-Physical Systems (affiliated to ESORICS'15), 2015.

- **Randomness in automotive-grade microcontrollers** is an important topic in cryptographic security, since all cryptographic designs ultimately rely on the quality of keys and nonces that are generated on the embedded devices. In our work (referenced below) we document an algorithm for extraction of random material from an automotive-grade microcontroller by reading from locations that are scattered in memory, trying to avoid portions that are loaded with deterministic material (e.g., the OSEK stack), in order to output a material with some fixed amount of entropy (estimated based on experimental data). We merge this source of entropy with the output from other devices that are present on the board: analog-to-digital

converters, bus performance counters, timers and clock calibration units. A patent application was submitted by Continental based on our research.

[A8] George Tipa (Continental Automotives), Bogdan Groza (UPT), Radu Ragobete (Continental Automotives), *Schema for generating true random numbers on automotive embedded devices*, European Patent Application, EP 14465511.5-1953/28.05.14, 2014

- **Vehicle access with mobile phones** is a topic that gained tremendous interest in the era of smart-phones and the myriad of hand-held gadgets. In spring 2015, we have been invited by Continental to contribute to the design of security protocols for such applications that facilitate access to the car via a mobile phone. The core ideas of our proposals were presented during Continental's Software conference in Regensburg:

  [A9] Adrian Radu (Continental Automotives), Bogdan Groza (UPT), *Security Concept for Smartphone Car access via NFC RF ID Device*, Continental Software Conference, Regensburg, 2015.

- **Remote vehicle diagnosis** is a topic that generated lot of interest on the industry's side, in the recent years there were dozens of patents applications in this area. This topic may not be so fruitful from a scientific point of view, however it is a topic of great interest for the industry with clear practical impact which nonetheless provides an interesting playground for security. Since 2013, we had several discussions with the industry on this topic and recently a patent proposal was submitted by Continental based on our research. Since there is a potential ongoing patent application, the results are not included in the thesis.

INVITED PRESENTATIONS AND RESEARCH SEMINARS. In addition to the presentations of published papers at various conferences, I have presented our work on automotive security in several other contexts: visiting research groups at other universities, and during research workshops or events organized by the industry. In 2013, I have given research talks at KU Leuven (June) and the University of Budapest (December). The same year I have been invited to give a presentation during the kick-start meeting (1st edition) of the BalkanCrypt event, currently at its third edition. In 2014, due to our involvement in patent applications with Continental Corporations I have been invited to a workshop on intellectual property rights to share some of our experiences. Recently (May, 2015) I have been invited to present our work at an industry's congress in Vienna held by Vector GmbH (a top manufacturer of diagnosis and simulation tools). These are summarized next.

[P1] Bogdan Groza, *LiBrA-CAN and beyond: Physically Unforgeable CAN (PSI-CAN) and Secure Automotive CAN (SeA-CAN)*, KU Leuven, COSIC, Research Presentation, 2013.

[P2] Bogdan Groza, *Security for Vehicular Buses: from Cryptography to Physically Unclonable Characteristics*, Budapest University (BME), Seminar Series on Advances in Telecommunications, Networking and Computing, 2013.

[P3] Bogdan Groza, *Experiences in bridging academic research in information security with*

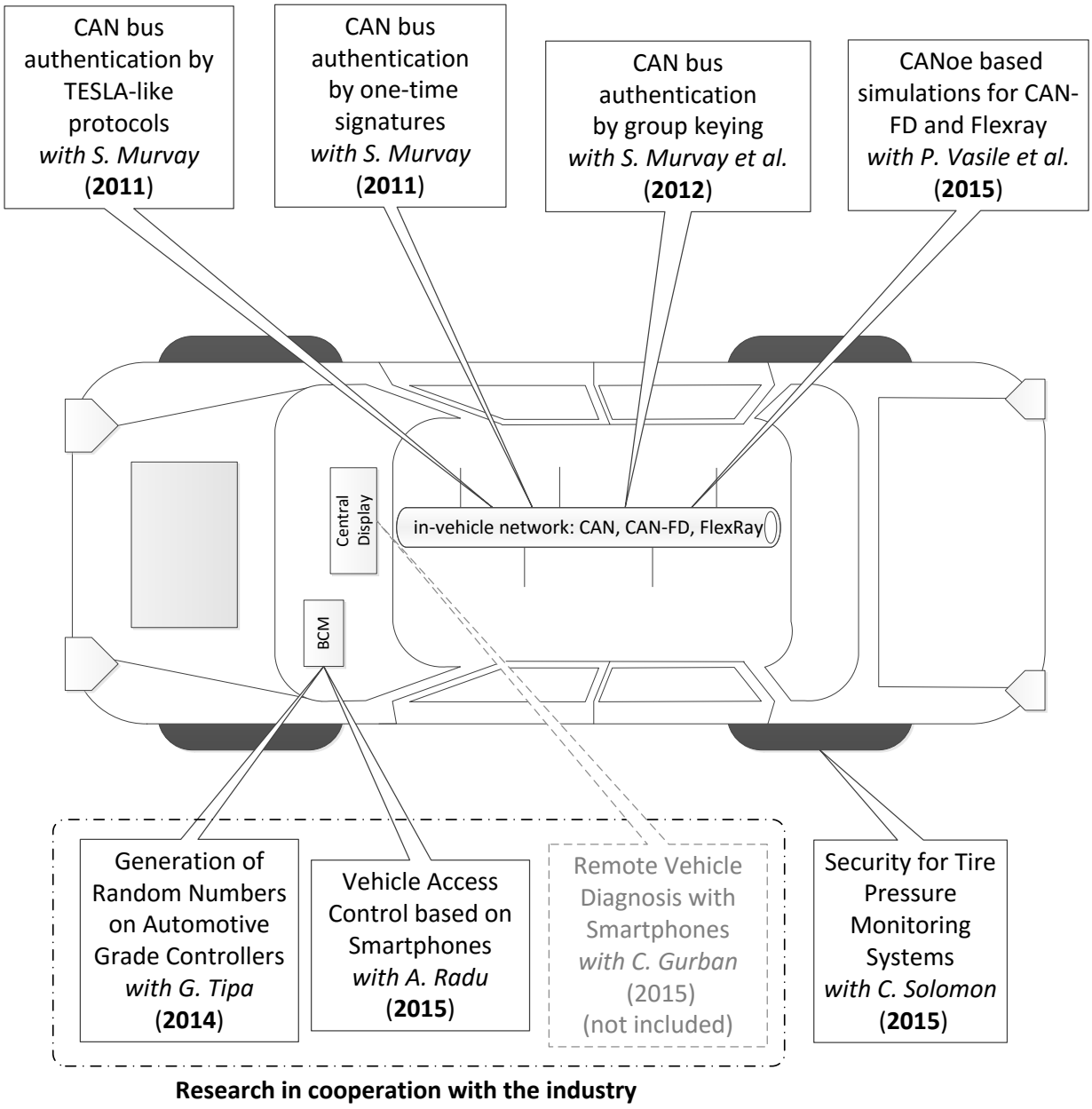


Figure 1.1: Overview of author's main research lines in automotive security (included in this thesis)

intellectual property and industry requirements, West University Timisoara, RO, Workshop on Intelectual Properties in ICT, 2014

[P4] Bogdan Groza, *Security for Vehicular Buses: from Cryptography to Physically Unclonable Characteristics*, BalkanCrypt, Sofia, Bulgaria, October, 2013.

[P5] Bogdan Groza, *In-vehicle security, bridging between academic research and industry requirements*, Vector Congress, Vienna, May, 2015.

RESEARCH GRANTS. Our research would not have been possible without a number of grants that have supported conference publications and participation to various events. Of great importance for the publication of our initial results in embedded systems security was research grant PN2-IDEI 940/2008-2011 focused on the security of industrial systems. In 2013, I became one of the two representatives of Romania in the management committee of COST Action IC1306 Cryptography for Secure Digital Interaction (funded by European Cooperation in Science and Technology) 2013-2017. Recently, I have been awarded a national research grant PN-II-RU-TE-2014-4-1501 for consolidating a team of young researchers in automotive security.

1.2 Research results in other areas of cryptography and systems security

While the research on automotive security is at the core of this thesis, this body of work would not have been possible without the experience and results from other areas of security and cryptography, e.g., theoretical cryptography, formal methods, security of mobile devices, etc. We list here only part of the results in these areas.

SECURITY BOUNDS FOR MULTIPLE INSTANCES OF CRYPTOGRAPHIC PRIMITIVES. Defining how security scales up over multiple instances of the same problem is a non-trivial topic, as it is commonly known that while a single instance of a problem may be hard, multiple instances do not always amplify hardness. Our work tries to build such security bounds for cryptographic puzzles, a moderately hard cryptographic problem. Part of the work was done while visiting Bogdan Warinschi at Bristol Cryptography Group (U. Bristol, U.K.). Two relevant papers have been published so far:

> [A10] Bogdan Groza, Bogdan Warinschi, *Client puzzles and DoS resilience, Revisited*, Designs Codes and Cryptography, Springer-Verlag, April 2013.
>
> [A11] Bogdan Groza, Bogdan Warinschi, *Revisiting difficulty notions for client puzzles and DoS resilience*, 15th Information Security Conference (ISC'12), Springer-Verlag, LNCS vol. 7483, 2012.

We do investigate two constructions that are trivial, but yet establishing the security bound for multiple instances appears to be problematic as erroneous bounds appeared in related work. The two constructions are the HashInversion puzzle, which consists in the partial inversion of a hash function (given $x'', H(x'||x'')$ find x') and the HashTrail puzzle which consists in finding an input to $H(r||\cdot)$ such that the result has a fixed number of trailing zeros. We then fix the difficulty bound as $\epsilon_{k,d,n} : \mathrm{N} \rightarrow [0,1]$ a family of functions indexed by parameters k, d (security and difficulty levels) and n (number of instances) such that for any adversary Adv it holds $\mathsf{Win}^{\Gamma/\mathsf{CPuz}}_{Adv,k,d,n}(q_{\mathsf{Gen}}, t) \leq \epsilon_{k,d,n}(q_{\mathsf{Gen}}, t)$ (where $\Gamma \in \{\mathsf{CS}, \mathsf{SS}\}$ denotes the game played either concurrent or sequentially). Additionally, we define a bound for the probability that the solving algorithm Find correctly finishes in at most t steps, denoted by $\zeta^{\mathsf{CPuz}}_{k,d,n}(t)$ is defined, i.e., $\zeta^{\mathsf{CPuz}}_{k,d,n}(t) = \Pr\left[\mathsf{Exec}^{\mathsf{CPuz}}_{\mathsf{Find},k,d,n}(t) = 1\right]$. The work proves that in the random oracle model, the Hash-

Trail puzzle has solving and difficulty bounds:

$$\zeta^{HT}_{k,d,n}(t) = \sum_{i=n,t} \binom{i-1}{n-1} \cdot \frac{1}{2^{nd}} \cdot \left(1 - \frac{1}{2^d}\right)^{i-n}, \qquad \epsilon^{HT}_{k,d,n}(t) \leq \zeta^{HT}_{k,d,n}(t) + \frac{1}{2^d - 1} + \frac{q^2_{\mathsf{Gen}}}{2^{k+1}}.$$

while for the HashInversion puzzle it holds:

$$\zeta^{HI}_{k,d,n}(t) = \sum_{i=n,t} [z^i] \left(z \cdot \frac{1 - z^{2^d}}{1 - z}\right)^n \cdot \frac{1}{2^{nd}}, \qquad \epsilon^{HI}_{k,d,n}(t) \leq \zeta^{HI}_{k,d,n}(t) + \frac{n}{2^d} + \frac{q^2_{\mathsf{Gen}}}{2^{k-d+1}}.$$

where $[z^i]P(z)$ denote the coefficient of z^i in the expansion of polynomial $P(z)$. These bounds contradict previous findings from [78] where for HashTrail the advantage was upper bounded by $\frac{q+n}{n \cdot 2^d}$ using the Markov inequality while the advantage of HashInversion was upper bounded by $\left(\frac{q+n}{n \cdot 2^d}\right)^n$. To ease computations, two approximations for the bounds are also provided:

$$\epsilon^{HT}_{k,d,n}(t) \leq \left[1 - \left(1 - \frac{1}{2^d}\right)^{t+1}\right]^n + \frac{1}{2^d - 1} + \frac{q^2_{\mathsf{Gen}}}{2^{k+1}},$$

$$\epsilon^{HI}_{k,d,n}(t) \leq \frac{1}{n}\left(\frac{t - n + 1}{2^d}\right)^n + \frac{n}{2^d} + \frac{q^2_{\mathsf{Gen}}}{2^{k-d+1}}.$$

Figure 1.2 shows a graphical depiction of the approximate bounds compared to the tight bounds of the previous two theorems and the bounds in [78]. These results are further complemented by a negative result on the efficiency of proof-of-works (PoW) against DoS attacks - a common application for client puzzles. We give a negative result showing that PoWs cannot provide DoS protection if $\pi_{Adv} > \pi_{\mathcal{C}} \cdot \theta^{-1}_{service}$ and there are no benefits in increasing puzzle difficulty to more than $d = \pi_{Adv}$ (here π_{Adv} and $\pi_{\mathcal{C}}$ denote the computational power of the client and the adversary while $\theta^{-1}_{service}$ is the inverse of the service time). Similar bounds are given for more complex PoW protocols that employ filtering or that use filters combined with an initial PoW.

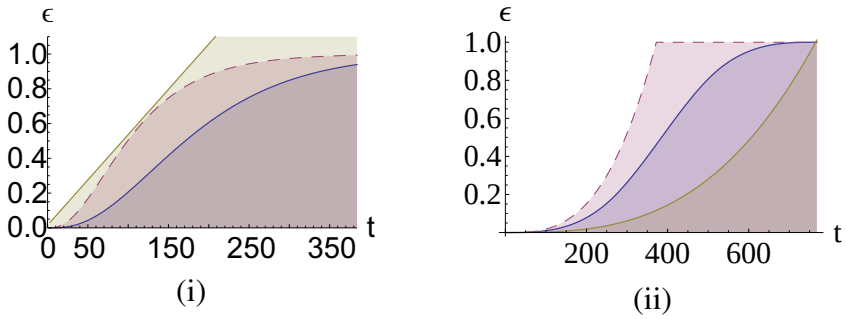


Figure 1.2: Graphical depiction of the approximate bounds (dotted line) vs. the tighter bounds and the bounds from [78] at $n = 3$, $d = 8$ for HashTrail (i) and HashInversion (ii) puzzles (based on our work from [A10])

AUTOMATED VERIFICATION OF CRYPTOGRAPHIC PROTOCOLS. This line of research was started collaborating with Marius Minea (UPT) in the framework of the AVANTSSAR (2008-2011) and SPaCIoS (2010-2013) FP7 research projects. Our contribution was on defining security models

for various attack classes that are not supported by standard protocol analysis: DoS attacks (i.e., resource depletion attacks) and guessing attacks (on low entropy values). This work represents the first attempt for automatic analysis of DoS resilience by the use of a model checker. In the second research project, our common work was focused on bridging the Dolev-Yao adversaries with control-systems, a largely unsolved topic that is opened for future research. We have published several articles on these topics:

[A12] Bogdan Groza, and Marius Minea. *A formal approach for automated reasoning about off-line and undetectable on-line guessing.* Financial Cryptography and Data Security. Springer Berlin Heidelberg, LNCS, pp. 391-399, 2010.

[A13] Bogdan Groza, Marius Minea, *Formal modelling and automatic detection of resource exhaustion attacks*. Proc. 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS'11), pp. 326-333, ACM, 2011.

[A14] Bogdan Groza, Marius Minea, *Customizing protocol specifications for detecting resource exhaustion and guessing attacks*. Proc. 9th International Symposium on Formal Methods for Components and Objects (FMCO'10), Springer-Verlag, LNCS vol. 6957, pp. 45-60, 2010.

[A15] Bogdan Groza, Marius Minea, *Bridging Dolev-Yao Adversaries and Control Systems with Time-Sensitive Channels*. Proc. 8th International Conference on Critical Information Infrastructures Security (CRITIS'13), Springer-Verlag, LNCS, 2013.

In brief, the main idea behind our work on modelling DoS resilience was to start from *cost-augmented* protocol descriptions. These are protocol specifications where the initial cost for all principals is zero, i.e., $\texttt{cost}(\texttt{P}, 0)$ holds for all principals in the *initial state*, and then each of the *transition rules* accounts for specific costs, i.e., $\mathcal{LHS}.\texttt{cost}(\texttt{P}, \texttt{C}_1) \Rightarrow \mathcal{RHS}.\texttt{cost}(\texttt{P}, \texttt{C}_2)$. Here $\texttt{cost}(\texttt{P}, \texttt{C}_1)$ is a fact denoting that the cost of principal P before the transition is C_1, and subsequently becomes $\texttt{cost}(\texttt{P}, \texttt{C}_2)$. Each operation of the principals (including Dolev-Yao capabilities) can be augmented by specific costs, e.g., the cost of encryption can be formalized as $\texttt{iknows}(\texttt{K}).\texttt{iknows}(\texttt{X}).\texttt{cost}(\texttt{i}, \texttt{C}_1)\ .\texttt{sum}(\texttt{C}_1, \texttt{c}_{\texttt{enc}}, \texttt{C}_2) \Rightarrow \texttt{iknows}(\texttt{enc}(\texttt{K}, \texttt{X})).\texttt{cost}(\texttt{i}, \texttt{C}_2)$. Subsequently, attack traces such as the one in Figure 1.3 can be found by the model checker.

Modelling guessing attacks was a more demanding procedure. We built upon intruder ability to *observe* or *control* oracles that take the target secret value as the input. In general, if the intruder control and observes such oracles, then guessing can be performed, i.e., $observes(O^f(s)) \wedge controls(O^f(s)) \Rightarrow guess(s)$. However, for sucessfull guessing, the intruder will also need to verify the output of the oracles for correctness. This would be a demanding procedure with state transitions, fortunately our approach benefited from the use of Horn clauses which results in a much more efficient procedure. Figure 1.4 shows an attack trace that is output after modelling the CHAP protocol.

As for bridging Dolev-Yao adversaries and control systems, this is a largely uncovered area which is difficult to approach. The main reason is that control theory usually works with continuous variables linked by differential equations, in contrast, protocol verification tools usually work on state transitions over a finite number of states. We do manage to bridge between the two over some simplified models, but there are much more aspects to be addressed by future research.

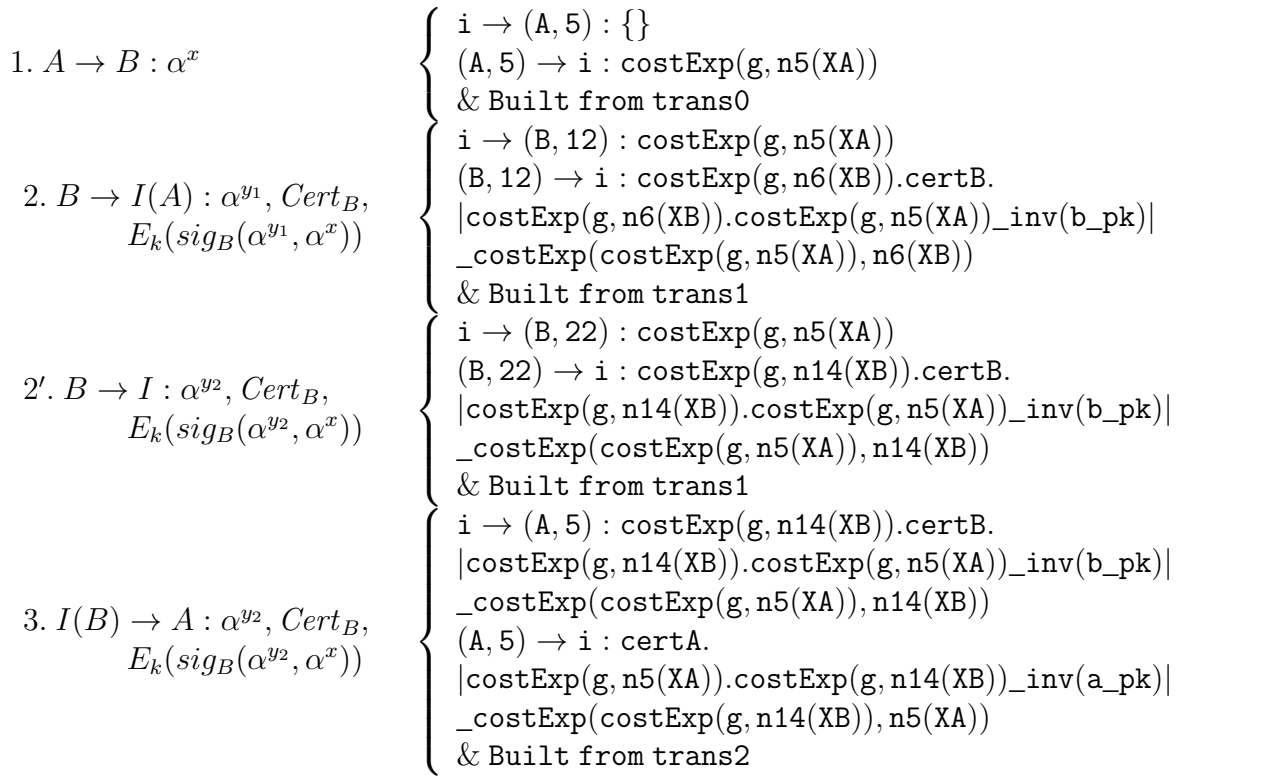


Figure 1.3: STS and the corresponding attack trace shown by CL-Atse (based on our work from [A14])

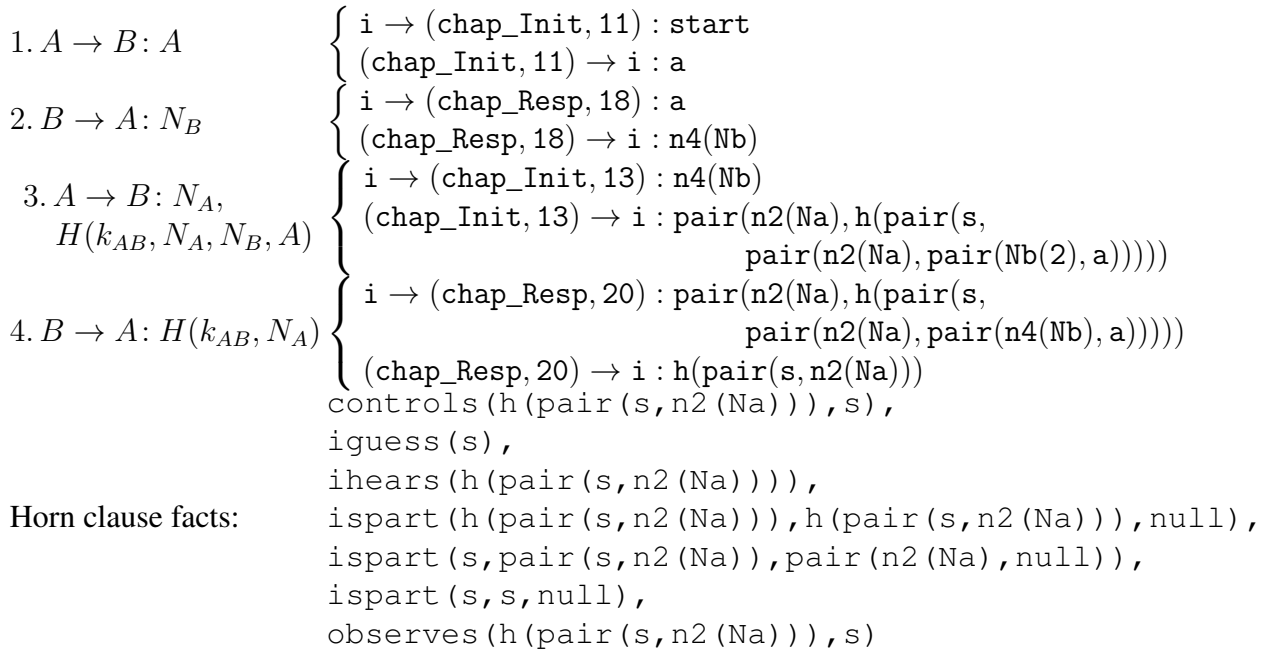


Figure 1.4: MS-CHAP v2 and the corresponding attack trace found by CL-Atse (based on our work from [A14])

SECURITY OF NETWORKS AND MOBILE DEVICES. Our research in this area are somewhat dispersed over several subjects, potential application of PoW (Proof-Of-Work) techniques against

e-mail spam [A18], using virtual coordinates to improve the effectiveness of PoWs [A19], security of wireless industrial devices [A17] and physical fingerprinting of mobile devices [A16]. Here, we will only present the results from the last of these works. The main result is in showing how mobile phones can be traced based on oscillator drifts that are recorded in ICMP packets when they connect over wireless networks (violating user's privacy over WiFi networks) . The idea of physical device fingerprinting by clock skews was first explored by Kohno et al. in [45]. Similarly, we used the clock skew as defined in [57] but this time for ICMP packets issued by mobile phones. Specifically, we let $x^i = \mathcal{T}_{\mathcal{C}}^i - \mathcal{T}_{\mathcal{C}}^0$ the difference in time between the i^{th} message and the first message (as reported by the target) and $y^i = \mathcal{T}_{\mathcal{R}}^i - \mathcal{T}_{\mathcal{R}}^0 - (\mathcal{T}_{\mathcal{C}}^i - \mathcal{T}_{\mathcal{C}}^0)$ as the observed offset. Now, if a smartphone p has received n ICMP timestamp requests, then let $\mathcal{O}_p = \{(x^i, y^i), i = \overline{0,n}\}$ be the offset for the smartphone p, and τ_p the clock skew of p, which will be the slope of the points in $\mathcal{O}_p$. To compute this slope, we follow the procedure from [45] by identifying the line $ax + b$ for which it holds $y^i - (ax^i + b) >= 0, i = \overline{0,n}$ and which minimizes the value $\min\{\sum_{i=0}^{n} y^i - (ax^i + b)\} = \min\{\sum_{i=0}^{n} y^i - a\sum_{i=0}^{n} x^i - nb\}$ (this is done by linear programming). Figure 1.5 shows the offsets and Figure 1.6 shows the clock skew as obtained from 4 distinct mobile phones that were subject to our experiments.

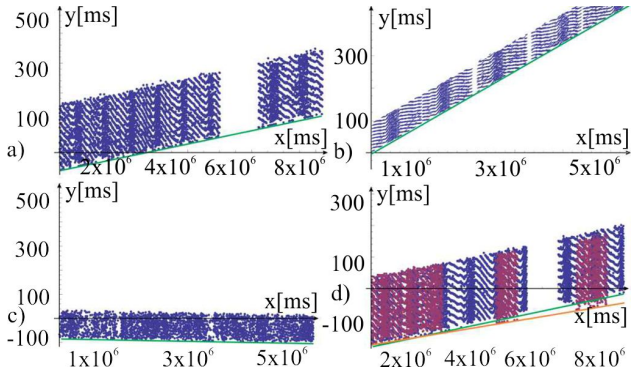


Figure 1.5: Clock offsets for: a) Samsung Google Nexus S; b) Motorola Motoluxe; c) Samsung Galaxy Mini S5570; d) two Samsung I9000 Galaxy S (based on our work from [A16])

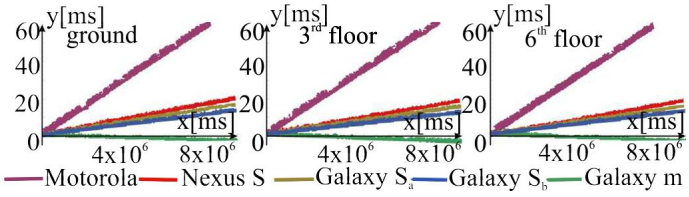


Figure 1.6: The clock skew recorded on 3 different access points (based on our work from [A16])

[A16] Marius Cristea, Bogdan Groza, *Fingerprinting Smartphones Remotely via ICMP Timestamps*, Communications Letters, IEEE, vol.17, no.6, June 2013.

[A17] Marius Cristea, Bogdan Groza, Mihai Iacob, *Some Security Issues In SCALANCE Wireless Industrial Networks*, Proc. 6th International Conference on Availability, Reliability and Security (ARES'11), IEEE Comp. Soc., pp. 493 - 498, 2011

[A18] Marius Cristea, Bogdan Groza, *Augmenting a webmail application with cryptographic puzzles to deflect spam*, In New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on, pp. 1-5. IEEE, 2011.

[A19] Marius Cristea, Bogdan Groza, *Provable Synthetic Coordinates for Increasing PoWs Effectiveness Against DoS and Spam*, In Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom), pp. 809-810. IEEE, 2012.

Another result in this area was published with Rene Mayrhofer (University of Upper Austria, Linz) and consists in a method for secure pairing of mobile devices based on accelerometer data, this method helps in associating devices based on environmental data (in contrast to the unhandy use of shared secrets when establishing a connection). Our procedure works in the expected way by comparing the values extracted from the accelerometer with a fixed threshold but improved on this by taking these values heuristically based on their distance to the threshold. Figure 1.7 shows the drift probabilities on two Wii accelerometers shaken together and Figure 1.8 shows the results after preprocessing from two accelerometers.

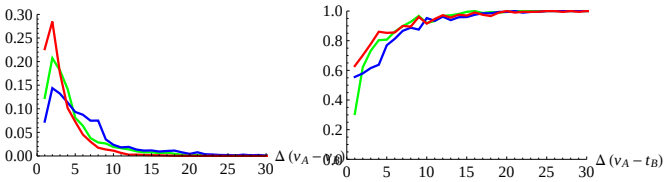
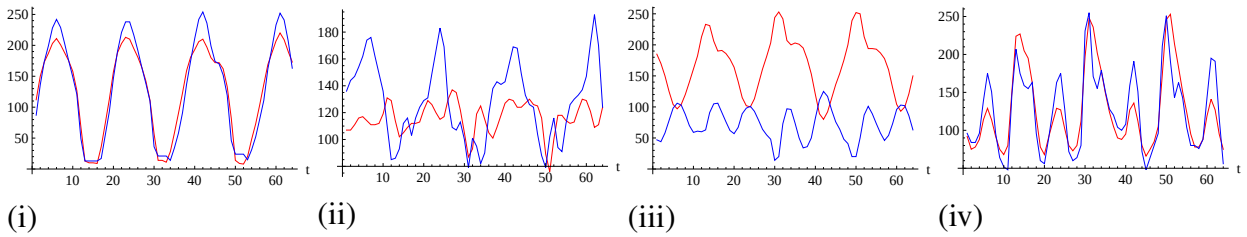


Figure 1.7: Drift probability from two WiiMotes shaken together (left) and success probability for different thresholds (right) (based on our work from [A20])



(i) (ii) (iii) (iv)

Figure 1.8: Example of data collected on the three axes (i), (ii), (iii) and the result after preprocessing (iv) (based on our work from [A20])

[A20] Bogdan Groza, Rene Mayrhofer, *SAPHE - Simple Accelerometer based wireless Pairing with Heuristic trees*, Proc. 10th International Conference on Advances in Mobile Computing and Multimedia (MoMM'12), ACM, 2012.

OTHER PRESENTATIONS AND RESEARCH SEMINARS. Related to all of the above, I had several research presentations at national events or foreign universities which are outlined below.

[P6] Bogdan Groza, *Invited Presentation: Current trends and challenges in cryptography*, National Hacking Event Defcamp, Timişoara, Romania, 2013.

[P7] Bogdan Groza, *Research Presentation: Client Puzzles, DoS Resilience, Multi-instance (Mi) Security - Revisiting Difficulty Notions*, KU Leuven, COSIC, Leuven, Belgium, 2013.

[P8] Bogdan Groza, *Research Seminar: Resource exhaustion attacks: formal verification and cryptographic countermeasures*, Upper Austria University of Applied Sciences, FH Oberösterreich in Hagenberg, Linz, Austria, 2012.

[P9] Bogdan Groza, *Research Seminar: Modelling of guessing and resource exhaustion attacks*, University of Bristol, Cryptography & Security Group, Bristol, UK, 2011.

[P10] Bogdan Groza, *Invited Presentation: Protocol vulnerabilities in practice: causes, modeling and automatic detection*, Romanian Cryptology Days, Bucharest, Romania, 2011.

OTHER ACHIEVEMENTS. For a complete list of publication and works please refer to the author's website (*http://www.aut.upt.ro/~bgroza*).

Chapter 2

Background on automotive security

In the past decade the automotive industry grew beyond margins that were predicted in the past. Current estimations suggest the number of vehicles in the world to be around 1 billion. Besides this significant growth, an important change in the building paradigm occurred: vehicles changed from mere mechanical devices to complex electronic systems where embedded devices are employed to perform some of the most exquisite tasks required for safety (braking, stability control, etc.), diagnosis or even user entertainment.

Regular modern vehicles incorporate up to a couple of dozen ECUs while luxury or commercial vehicles have a much higher number, e.g., a premium vehicle such as the 2004 model of VW Phaeton already had 65 ECUs and 11.136 electrical parts in total [49]. Years ago, Broy [9] already placed the expected number of ECUs at 70 with more than 2000 software functions. Needless to say, all these ECUs need to communicate and whenever communication occurs, security needs to be put in place since otherwise malicious adversaries can manipulate information that further affects the safety and security of drivers and passengers.

The automotive industry was no stranger to security issues, e.g., car theft and car tuning are common practices for many decades, but today implications can reach well beyond these common burglaries. Recent research showed that adversaries can mount attacks with catastrophic consequences: disable brakes, kill engine, remote car start, engage unevenly or release brakes without driver knowledge, etc., all these attacks are described in [46], [17], [3], [41] and other works. The socio-economic impact of such attacks can be disastrous as it can lead to human loses and high economic costs, especially in the era of cyber-terrorism. To confirm this, recent news shows that the U.S. vehicle industry will start cooperation with military hackers to improve vehicle security following a demonstration that showed how hackers can take control over some of the recent-most models (launched in 2010) of Toyota Prius and Ford Escape [67]. The attack only requires a hacker to insert a miniature device in the car that connects to the in-vehicle network – the simplicity of doing along with the disastrous consequences makes this a very serious issue. Less than a year later, hackers took remote control of a Jeep car, disabled the breaks and smashed it into a ditch [26]. This resulted in Jeep making a recall for software updates on 1.4 million vehicles [25].

All these attacks are possible because the communication buses inside cars are not protected by cryptographic/security protocols. Since security on vehicular buses is almost absent, the importance of solving this problem is unquestionable, and similar to the case of wireless or wired networks (e.g., Internet, LANs, WANs, etc.) there is no question that cryptographic security is the way to alleviate this problem.

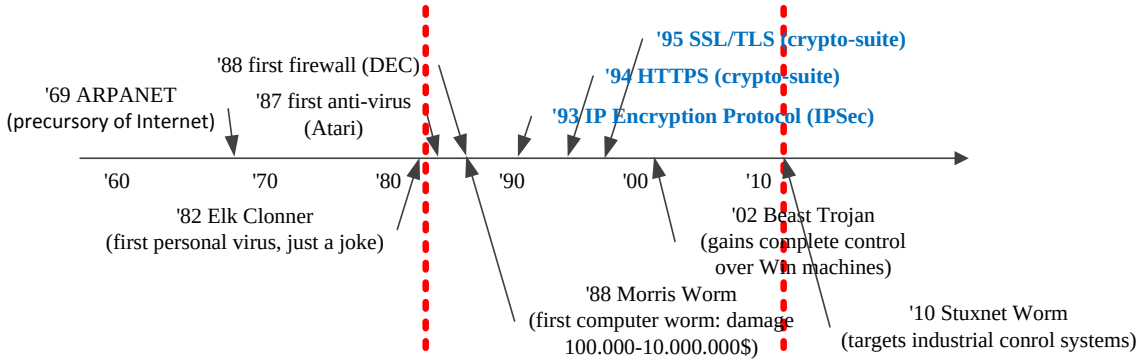


Figure 2.1: Evolution of some security threats and defence mechanism in the computer industry

2.1 Security through isolation, an illusion

Security through isolation is the first option which one considers. This was (perhaps still is) the vision for most automotive manufacturers. Isolating all in-vehicle buses, e.g., CAN, from the outside world is apparently easier to adopt. However, there are several problems with this option:

1. 100% isolation is never achievable, e.g., it is projected that cars will need to receive more and more updates from the outside thus there will always be some outside connectivity,

2. may not fit the very nature of cars, e.g., cars are frequently left in unattended park slots or in less trusted repair facilities maybe hundreds of miles away from the owners home in case of impairments,

3. may not work for the medium-long term due to increase connectivity demanded by new paradigms, e.g., self-driving cars, vehicle-to-vehicle communication, etc.

It seems unlikely that one can give guarantees for the in-vehicle network to remain isolated. The image of security through isolation deceived its followers many times. Before the 90s, computers were assumed by many to be secured by isolation. Industrial devices were assumed to be secured by isolation before the Stuxnet attack in 2010.

Figure 2.1 is suggestive for the evolution of some security threats and defence mechanism in the computer industry.

We can make the same predictions for in-vehicle networks. If in-vehicle networks are 10-20 years behind computer networks, then we can expect the first in-vehicle malware to arrive until 2020. This clearly leads us to the second option which is: devise security for in-vehicle buses. It is harder to adopt for the short term, but has clear advantages for the long term future. Figure 2.2 is suggestive for the evolution of in-vehicle networks and reported attacks.

2.2 In-vehicle connectivity (networks)

Security problems generally arise from a connected world. It is all the same if we are talking about computers which get affected by malware or viruses once they are connected to the outside, be

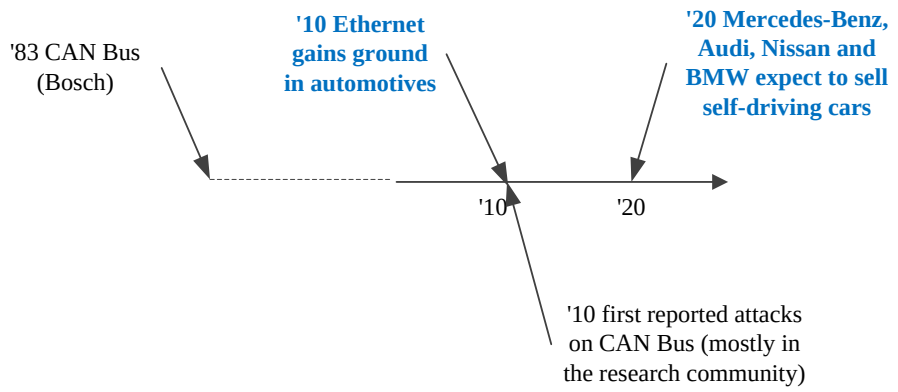


Figure 2.2: Evolution of in-vehicle networks and reported attacks

it on-line via a network connection or off-line by plugging some malicious device. Cars are no exception and all the security issues come from the communication interfaces, i.e., the in-vehicle network.

The data communication interfaces inside a vehicle evolved from several wires to a maze that gathers various wired of wireless protocol suites and connects inputs from physical components, ECUs or devices (possibly insecure) handled by humans. There is everything, starting from low speed wired or wireless interfaces that barely exceed a dozen kilobytes per second, up to high speed interfaces that exceed dozens of megabytes per second (much more is projected for the future). Figure 2.3 gives a suggestion of these communication interfaces. This image offers only an intuition as the network architecture from a high-end vehicle is much more complex. To the best of our knowledge, a complete real-world network architecture for a high-end vehicle is not available in the public domain, so it would have been impossible to reproduce one here.

In what follows, we briefly describe the communication interfaces that are regularly present in cars:

- *LIN (Local Interconnect Network)* is a serial communication interface designed to be a cheap alternative for CAN. It has a master-slave architecture and reaches speeds of up to 20 kbps. It is intended to assure connectivity between various peripheral sensors and actuators for doors, windows, etc. Due to its limited bandwidth and also due to the limited computational power and memory of the devices that usually operate on LIN it is unlikely that security (cryptographic) will ever be designed for this bus.

- *CAN (Controller Area Network)* is the workhorse behind most of the existing in-vehicle networks. Designed by Bosch since 1983, it proved to be so successful that today it is present in virtually every car. It is a two wire broadcast bus as shown in Figure 2.4. The structure of the CAN frame consists in the arbitration field (referred as the identifier ID), a 6 bit control field, 0-64 bits of data, a 15 bit CRC and a 2 bit acknowledgement field. Additionally 1 bit marks the start of frame and 7 bits mark its end. This structure is suggested in Figure 2.5. Arbitration is based on the identifier ID which has 29 bits in extended frames and 11 bits in standard frames. The winner is determined based on the state of a particular bit, namely recessive bits

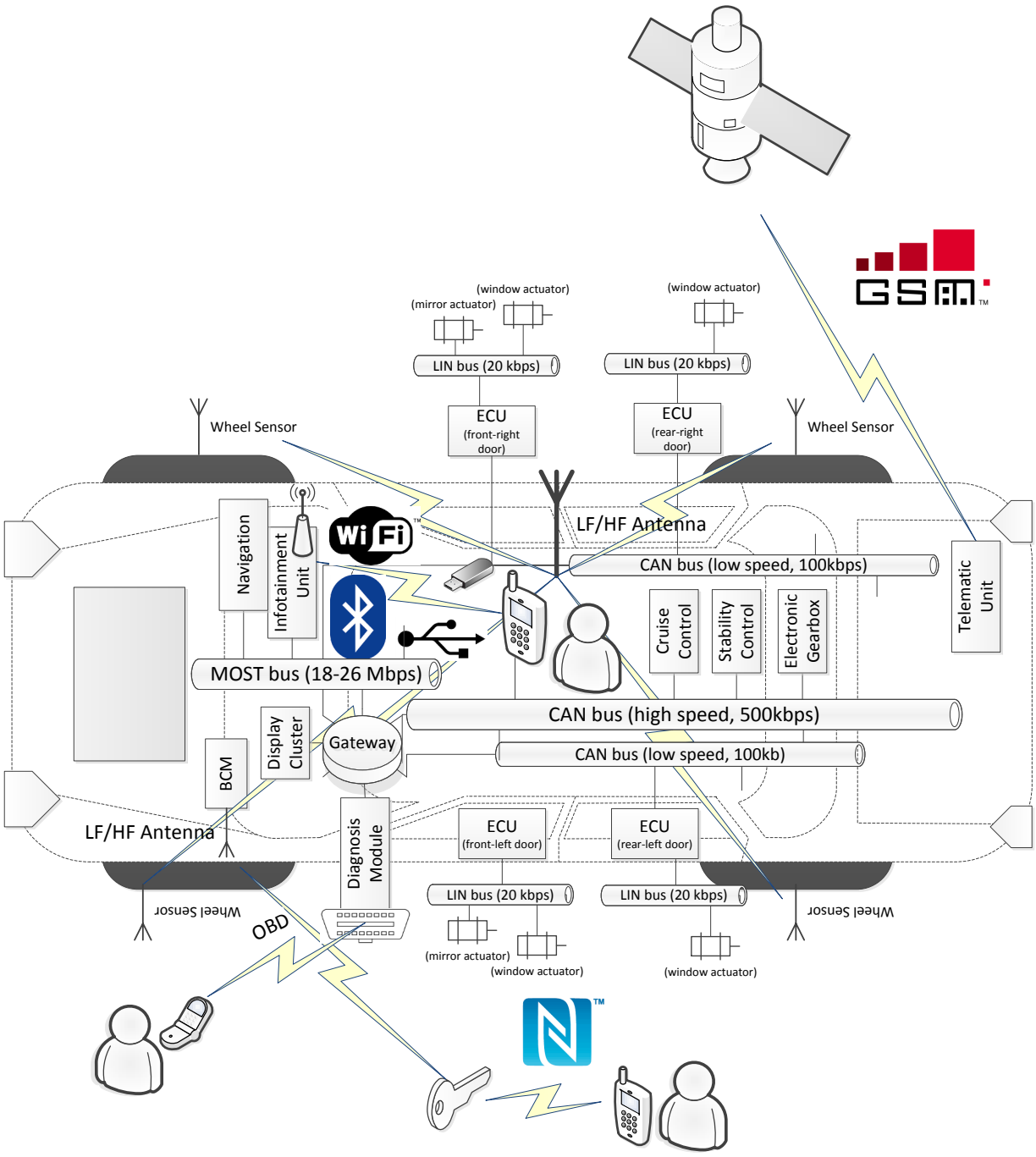


Figure 2.3: Suggestive depiction of some in-vehicle communication interfaces

(value 1) are overwritten by dominant bits (value 0). If needed, all nodes can start to write a message at the same time on the bus, but, whenever a node writes a recessive bit and reads a dominant one it means that it lost the arbitration and will stop, otherwise it can continue. Arbitration based on message priority is a relevant feature of the bus, but it should be noted that it also opens the door for DoS attacks if one malicious node starts to write packets with

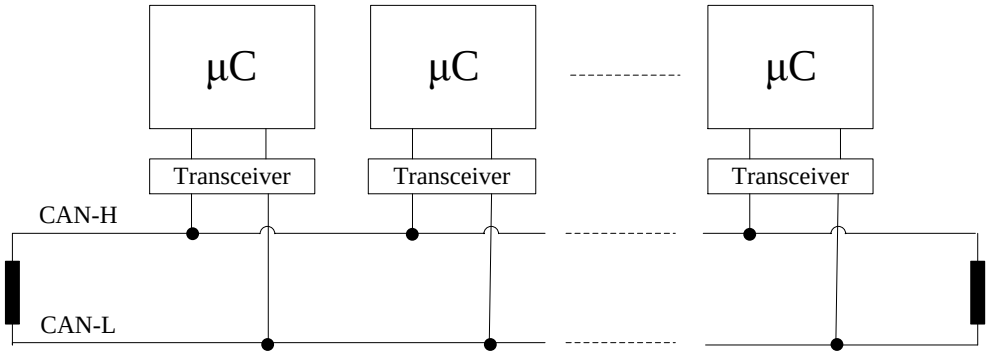


Figure 2.4: Topology of CAN bus

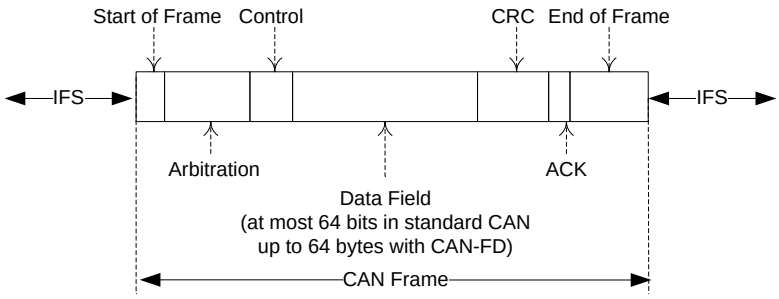


Figure 2.5: Structure of a CAN frame

low value IDs that will always win the bus. After each 6 consecutive bits of identical values a stuffing bit of different value is added. The body of a message can have at most 8 bytes and is followed by a 15 bit CRC. In the worst case, a frame can have 154 bits out of which only 64 bits are of actual useful information. Thus, the overhead is high from the basic design of the protocol, in the worst case exceeding 50%. But this is needed to achieve reliability as mentioned before. Two kinds of CAN nodes are commonly available on the market: fault tolerant low-speed nodes which operate at 125kbps and high-speed nodes that work up to 1Mbps. Usually, inside a vehicle several CAN sub-networks exists (some working at low speed, other at high speed) and these networks communicate with each other over gateways. The computational power and memory resources of CAN capable devices ranges from low to very high (multi-core on 32 bit, etc.). Implementing security on this bus is feasible and needed. Section 3 of the thesis contains results on this matter. The main limitation of CAN is its bandwidth upper-bounded at 1Mbps which may be insufficient for current needs. However, its successors (CAN-FD or FlexRay) can benefit from the same security mechanisms that we discuss in the next sections since these mechanisms are not necessarily bound to the network layer.

- *CAN-FD (CAN with Flexible Data-Rate)* was designed as replacement for CAN which offered insufficient bandwidth for modern needs. The improvement of the classical CAN is in

that it allows a distinct (higher) data rate during the data field than during arbitration (lower). This leads to a payload that increases from the 64 bits in CAN frames to 512 bits, i.e., 64 bytes. From a security perspective this nicely facilitates the deployment of cryptography. Current transceivers allow a bandwidth of 2.5 Mbps. Some view CAN-FD as an intermediary until complete replacement of CAN like buses with Flex-Ray or Ethernet which can reach speeds of 10-100 Mbps.

- *FlexRay* was designed by a consortium of automotive producers (Bosch, BMW, Daimler AG, etc.) as a faster and more reliable alternative to CAN. However, it is likely that the inherent expenses of this bus (both in production and software development) have slowed its adoption in practice, the bus is present in many vehicles but seems still far from replacing CAN. Current rumours even suggest that it will not be the case for FlexRay to replace CAN but it is rather that Ethernet will bring both CAN and FlexRay to an end. The main advantage of FlexRay over CAN is of course its bandwidth that can reach up to 10Mpbs. But this is not the only advantage. Perhaps a more important one is that the bus assures a time-triggered communication removing limitations of the CAN bus in front of flooding (innocuous or not) with messages of high priority.

- *Ethernet* is slowly but steadily arriving in cars. There are many advantages that this bus carries, besides the speed, the most important is its reduced cost due to large scale production of Ethernet capable devices. The bus is not yet wide spread in cars (e.g., present in some models for OBD, display clusters, etc.) and it was not yet subject of our research, but it is a promising layer for future work. In the past, Ethernet was never seen as a core bus for automotive data communications, but this image appears to be drastically changing.

- *MOST (Media Oriented System Transport)* is a bus designed for audio-video communication. Its first release was capable of 23Mbps and recent versions allow up to 6 times this speed. Still it was not intended for safety-critical applications and cannot be easily seen as a competitor for CAN or FlexRay. It is likely that Ethernet is also the competitor and potential replacement for MOST.

- *Radio frequency (LF & HF)* interfaces are available in various parts of modern vehicles starting from wireless keys to wireless sensors. Both wireless keys and sensors are of interest to us. While the vast majority of sensors inside the vehicles are connected by cables, one notable exception to this rule are the sensors located inside the tires of the vehicle. We discuss security issues related to these devices in a forthcoming section dedicated to our research.

- *Bluetooth, WiFi, NFC* all come from the need to connect the car with smart gadgets such as phones, tablets, etc. This opens an entire new vista for usability but also a distinct security perspective. This comes from the fact that none of the user-held devices can be assumed to be trusted (un-compromised) and they can open road to future unexpected attacks.

2.3 Challenges, why is it difficult to design in-vehicle security?

Assuring security in automotive systems is a difficult task due to at least three obvious challenges:

1. dealing with a constrained environment where resources are limited, a rather classical problem in designing security,

2. dealing with low cost margins on the side of the producers, not a new constraint but a somewhat spectacular one as producers seem to be careful even on spending an extra dollar for a car that costs thousands,

3. bridging academic research with the industry as it is clear that while the automotive industry has vast know-how in designing vehicles, the security know-how resides outside, in the research community.

The fact that micro-controllers are constrained devices is commonly known, automotive grade embedded devices are no exception. This puts the problem of security under constraints from memory, processing power, storage size and communication bandwidth. The recent-most high-end automotive devices have two 32 bit cores that run at 100-300 Mhz (e.g., Infineon TriCore TC1797 and its siblings is a well used micro-controller in the automotive industry [39]), but it never happens that such devices are the only devices inside cars and one must always expect the presence of low-end devices that have one 8 or 16 bit core that runs at 20-40 Mhz (e.g., Freescale S12X is a prominent example from the automotive industry [22]).

Designing cryptographic security in such environments is challenging. To intricate things further such devices are common subject of specific attacks such as side-channel attacks which makes the problem even harder to solve [44]. Dealing with all these can be done by increasing the performance of devices and protection layers but this also increases the costs and here we are facing the second challenge: low cost margins. The automotive industry tries to keep production costs as low as possible. This difficulty can be reimbursed by using efficient, lightweight cryptographic designs which are the focus of the cryptographic community for decades and there are no doubts that suitable solutions exist but the challenge is in identifying, adapting and implementing them in automotive grade devices. This leads us to the third difficulty which is in bridging academic research with the industry. The automotive industry is usually opaque with its specifications. However, hundreds of security mechanisms that assure our safety over complex network (such as the Internet) rely on open standards that proved their effectiveness over the years (IPSec, SSL/TLS, etc.). There is no doubt that sooner or later the automotive industry will open at least a part of their specifications. In part, this already happens with standards such as AUTOSSAR - AUTomotive Open System ARchitecture [2]. An open model that heavily relies on feedback from the academic community will also reduce the production costs.

2.4 State-of-the-art, overview of recent proposals

In the recent years, several monographs appeared on the subject of embedded security in cars [48], [99] and security in vehicular networks [34], [38], [66] but the large majority of solutions are new and therefore not properly analysed (security needs careful analysis over the years) and there are several uncovered areas (e.g., in-vehicle networks in particular CAN bus, Flexray, TPMS [41], etc.). We discuss next along the lines of the State-of-the-Art.

Currently there are no security mechanisms on communication buses inside cars (e.g., CAN, FlexRay) or over most of the wireless sensor interfaces (e.g., TPMS). The industry is starting to change this perspective, a solid proof for this is the fact that the industry standard AUTOSSAR has started a few years ago to standardize cryptographic interfaces [2]. There are also attempts to implement and cover security solutions by patents, e.g., a TPMS security patent was developed by one of our former students employed by Continental [89]. These are only the first steps, as

there are yet no protocols designs for key-exchange, tunneling, etc. commonly agreed by industry representatives, tested and deployed on automotive buses.

The only cryptographic security mechanisms inside cars are the ones from subsystems such as electronic immobilizers, digital tachographs [1], [24] (that embed both public and secret key cryptography), etc. and some security mechanisms that are inherited from the network layers, e.g. bluetooth, 4G, etc. These mechanism may have themselves flaws: bluetooth was seldomly found vulnerable [33], [15], the A5/1 cryptographic algorithm employed by GSM has severe flaws and is breakable [32], NFC is subject to a plethora of attacks: eavesdropping [36], DoS [53], [58], phishing [58], relay attacks and data modification [36], while WiFi communications for vehicle-to-vehicle communication raises even more concerns [62], [43]. Another key aspect of automotive security is that even issues that are addressed for decades seem to be far from solved. Car theft is still a concern despite clever electronic immobilizers [86], [96] while modern NFC [13], [79] Bluetooth [18] and RFID [98] solutions are also prone to attacks, mileage modification [85] is commonly done by garage owners despite protections inside ECUs. This makes the subject even more interesting, the security analysis of such devices has not yet endeavour many academic efforts and would be of great scientific interest before these devices will become more exposed to hackers and intruders.

Moreover the interaction between such mechanisms and vehicle sub-systems is poorly if at all studied, some works clearly acknowledge that these interfaces can serve as attack entry points for cars [80], and a recent remarkable example from [17] shows how a 4G communication channel can be exploited from hundred miles away to subvert communication of passengers inside a car and how to eavesdrop communication or compromise ECUs from Bluetooth connections. The impact of adding security mechanisms to the real-time nature of automotive applications has only recently start to be evaluated [54]. The performance of automotive embedded devices used for achieving cryptographic security is not an intensely studied subject. Results published so far are fragmented as they are contained in papers which have adjacent topics as a main subject: performance evaluation of cryptographic primitives [59], implementations of various primitives with optimisations for a particular platform [55]. Previous research was also concerned with designing implementations compliant with the AUTOSAR crypto security module [11].

All of the previous, prove that security in automotive devices and networks is a serious and challenging problem that requires more research efforts.

SECURITY ON VEHICULAR BUSES, PROPOSED PROTOCOLS. We now enumerate some of the recent proposals (part of this synthesis is also present in one of our publications [30] but here we update it according to the most recent advances, including some results from [95]). In a forthcoming section, some of these proposals are revisited from a key allocation perspective which is decisive for the performance evaluation that follows. The following discussion is also summarized in Table 2.2.

1. *Voting schemes* were introduced by Szilagyi and Koopman in [82], [83], [84]. Their scheme is intended for generic time-triggered communication such as TT-CAN, FlexRay, etc. The core part of the protocol relies on the classical paradigm of sharing keys between each sender and receiver then authenticating packets on a one MAC per receiver basis. Further, to make it feasible to embed the MACs in a single frame, the tags are truncated and concatenated (e.g., 3 MACs each of 8 bits are fitted at the end of a single frame). The communication is time triggered, each receiver releasing his message and his vote on previous messages in fixed time slots. Both the new message and his vote, along with all previously received messages, are authenticated under the same array of MACs to other receivers. The scheme appears to be a trade-off between computational time, authentication delays and bandwidth in order to fit the

authentication bits in one frame. Indeed, if the frame would be larger, and the sender could fit more MAC bits in each frame, then authentication could be done at once within a single frame without needing to wait for the votes of the other nodes. This would improve both on delays (as nodes will not need to wait for the vote of other nodes) and computational power since, indeed, the nodes that subsequently vote are re-authenticating messages that were previously authenticated with a small amount of bits. The procedure leads to a drawback as stated in [83]: for frames that are lost, the receive history of the nodes does not match and authentication will fail for these frames. As suggested in [83] this can be fixed by adding additional bits for lost packets, but sufficient votes from other nodes would still be required to deem the frame authentic. The main problem with the scheme is that it requires nodes to be present and vote on the authenticity of messages. Additionally, a message needs to accumulate a sufficient number of votes which introduces additional delays and the receive history of the nodes has to match. These restrictions seem not to be appropriate for in-vehicle networks.

2. *TESLA* like protocols proved to be highly effective in sensor networks [65], [64] and so far are the most efficient alternative for assuring broadcast authentication with low-cost symmetric primitives, e.g., Message Authentication Codes (MAC). Implementing this protocol on the CAN bus was considered by us in [29]. There is a significant drawback in adopting it for in-vehicle networks and this is due to the authentication delay which is always present in such protocols. The problem is not only that messages can be authenticated with some delay, e.g., usually 1–10 ms, but the node has to buffer all messages and authentication tags received in this time slot for subsequent authentication (this raises memory concerns). However, insofar TESLA is the only way for assuring full broadcast authentication without more expensive public-key cryptographic primitives, e.g., digital signatures. The main objective of our work in [29] was to determine a lower bound on these delays and establish some trade-offs. Delays in the order of milliseconds, as shown to be achievable in [29], are satisfactory for many scenarios, but such delays do not appear to be small enough for in-vehicle communication. To reduce the delays, one can use a bus with a higher throughput, more computational power and better electronic components (e.g., oscillators), but this will greatly increase the cost of components, nullifying in this way the cost effectiveness of CAN.

3. *CANAuth* is a scheme that proposes the use of an ID-oriented key allocation [91]. The protocol has the merit of following in great detail the specifications of CAN, its security is specifically designed to exactly match the requirements of the CAN bus. In particular, CANAuth is not intended to achieve source authentication as the authentication is binded to the message IDs and messages may originate from different sources which will be impossible to trace. This fits the specification of CAN which has a message oriented communication. However, a first issue is that the number of CAN IDs is quite high, in the order of hundreds (11 bits) or even millions in the case of extended frames (29 bits) and storing a key for each possible ID does not seem to be so practical. For this purpose, in [92] a clever solution is imagined: the keys are linked to multiple ID codes using masks, which greatly reduces the number of keys. But still, this leads to some security concerns. Traditionally, keys are associated to entities to ensure that they are not impersonated by adversaries, but by associating keys to messages the identity of the sender can be no longer verified. For example, any external tool (assume On-Board Diagnostics (OBD) tools which are wide-spread) that is produced by external third parties will have to embed the keys associated for each ID that it sends over or even just listens to on CAN. It is thus unclear which keys can be shared with different manufacturers

and how or what the security outcomes of this are. Obviously, if a third party device is easy to compromise (even an innocuous one such as passive receiver) then all the IDs which it was allowed to authenticate are equally compromised. Thus, sharing the key with nodes that are allowed to receive a particular ID (in order to be able to authenticate the messages) exposes the security key to nodes that can be potentially malicious (e.g., a corrupted diagnosis tool). However, the ID-oriented allocation nicely fits the specification of CAN. Moreover, by adding a single authentication tag to each of the broadcast message this scheme provides a baseline for efficiency as it leads to a simple one-tag-per-message authentication. As proposed in [91] the protocol was intended for CAN+, a variation of CAN that currently does not exist in practice. But the protocol can be ported as is on any other layer, e.g., CAN or CAN-FD.

4. *MaCAN* is proposed in [35]. The protocol has the merit of being a realistic proposal, it employs shared keys between nodes and CBC-MAC based authentication tags. There are not enough details on how to share the keys between the nodes, except for the fact that these should be shared in a pair-wise manner (this is not suitable for higher number of nodes). The authors of MaCAN [35] suggest that nodes can be grouped under the same key if they share the same trust level which will lead to a reasonable number of keys, but no practical insights are given on how to decide the trust level. Subsequent formal analysis [10] found the protocol vulnerable to some attacks but easy to fix. One more serious security issue with MaCAN is that the use of a counter is avoided and freshness is assured by linking to a common time value preserved on the nodes. Needless to say, synchronization errors will always be present and the protocol appears to provide no protection against adversaries that can force such errors by delaying the synchronization packets from the time server. This means that replay attacks cannot be completely avoided in MaCAN. Moreover, messages are authenticated only for a fixed number of times, this leaves the possibility for an adversary to compromise frames that are missing the authentication tag.

5. *LiBrA-CAN* [28] proposes a more demanding key allocation procedure which mixes keys between groups of nodes (rather then sharing keys pair-wisely). The protocol is more intensive from a computational point of view, but it offers a higher security level in case when adversaries are in minority (a likely scenario for in-vehicle networks). Given that bandwidth proved to be the main limitation for in-vehicle networks (our results on TESLA-CAN advocate for this) it was a natural evolution to trade some computational power in exchange for some additional security. Besides mixing the keys between groups of nodes, LiBrA-CAN also makes use of a more advance 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. Our experimental results suggest that LiBrA-CAN is the best alternative for assuring security on CAN-like buses.

6. *CaCAN* [47] introduces a centralized view over the authentication process. In this protocol a central node verifies the authentication tag of each frame and if authentication fails, the frame is discarded with error flags. This procedure has the merit of requiring a single monitor node with higher computational power. However, an adversary that removes this node from the bus can take full control of the bus since there is no way for the other nodes to decide if a frame is authentic or not.

7. *Physical signal characteristics* is a recent development that we discuss in [60]. This cannot

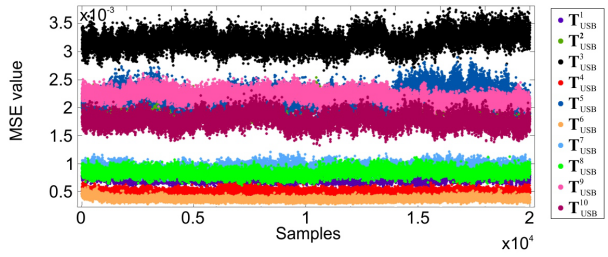


Figure 2.6: MSE separation values with a first transceiver fingerprint from 2×10^4 values (as depicted in [60])

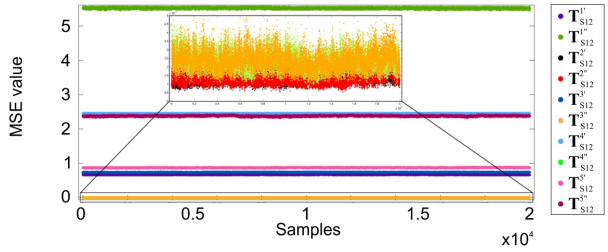


Figure 2.7: MSE separation values with a second transceiver fingerprint from 2×10^4 values (as depicted in [60])

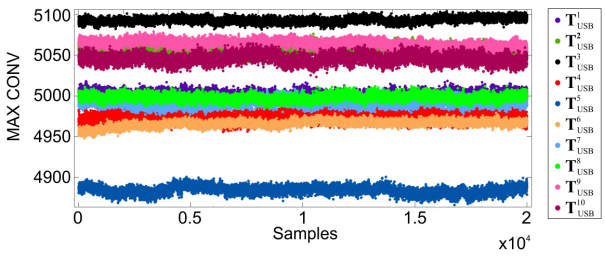


Figure 2.8: Convolution based separation values with a first transceiver fingerprint from 2×10^4 values (as depicted in [60])

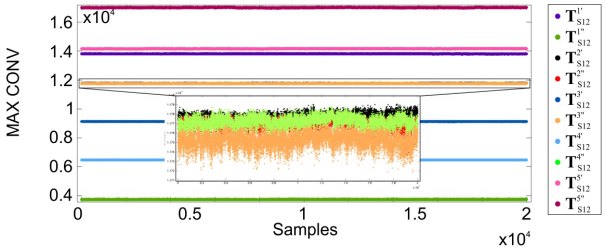


Figure 2.9: Convolution based separation values with a second transceiver fingerprint from 2×10^4 values (as depicted in [60])

be considered as part of the cryptographic approaches for assuring security on the CAN bus, a reason for which we do not include it in the forthcoming section dedicated to our results on assuring cryptographic security for in-vehicle networks. The procedure makes use of physical signal characteristics to identify CAN nodes. The main advantage of this procedure is that it does not require cryptography removing the problem of sharing keys or adding overhead in the CAN packets, moreover it is fully back-ward compatible. However, it also bears a signficant drawback: it is unclear what are the deployment costs of this solution and how easy would be to build a node that has almost similar characteristics to an existing one. We at least try to give a partial answer by showing that this solution may work. Our research in [60] shows that one can successfully distinguish between CAN nodes with good success probability by using some mathematical tools over the signal, e.g., low-pass filtering, mean square errors, convolutions, etc. Figures 2.6, 2.7, 2.8, 2.9 graphically depict this separation for a first and second transceiver based on means square errors and convolutions. Moreover, in case when two nodes are indistinguishable on a particular ID, we show that one can correctly make a distinction by using a distinct ID for the two nodes. This suggests that by clever allocation of the IDs for the nodes, one can build a network where the nodes are recognizable based on patterns of the physical signal. This solution requires more investigations and may be of interest to our future work as well. Table 2.1 shows the success rates in distinguishing between ten transceivers from S12 development boards, we use ✓as place holder to denote that during experiments there was no confusion between the two transceivers. The complete experimental results and further details can be found in our work [60].

Table 2.1: Identification rates for TJA1054T nodes with ID set to 0x000 (as depicted in [60])

Target	$\mathbf{T}^1_{\mathrm{USB}}$	$\mathbf{T}^1_{\mathrm{USB}}$	$\mathbf{T}^2_{\mathrm{USB}}$	$\mathbf{T}^2_{\mathrm{USB}}$	$\mathbf{T}^3_{\mathrm{USB}}$	$\mathbf{T}^3_{\mathrm{USB}}$	$\mathbf{T}^4_{\mathrm{USB}}$	$\mathbf{T}^4_{\mathrm{USB}}$	$\mathbf{T}^5_{\mathrm{USB}}$	$\mathbf{T}^5_{\mathrm{USB}}$
$\mathbf{T}^1_{\mathrm{USB}}$	1	✓	✓	✓	✓	✓	✓	✓	✓	✓
$\mathbf{T}^1_{\mathrm{USB}}$	✓	1	✓	✓	✓	✓	✓	✓	✓	✓
$\mathbf{T}^2_{\mathrm{USB}}$	✓	✓	0.908	0.876	✓	0.204	✓	0.029	✓	✓
$\mathbf{T}^2_{\mathrm{USB}}$	✓	✓	0.831	0.901	✓	0.118	✓	0.121	✓	✓
$\mathbf{T}^3_{\mathrm{USB}}$	✓	✓	✓	✓	1	✓	✓	✓	✓	✓
$\mathbf{T}^3_{\mathrm{USB}}$	✓	✓	0.999	0.991	✓	0.901	✓	0.300	✓	✓
$\mathbf{T}^4_{\mathrm{USB}}$	✓	✓	✓	✓	✓	✓	1	✓	✓	✓
$\mathbf{T}^4_{\mathrm{USB}}$	✓	✓	0.527	0.934	✓	0.029	✓	0.900	✓	✓
$\mathbf{T}^5_{\mathrm{USB}}$	✓	✓	✓	✓	✓	✓	✓	✓	1	✓
$\mathbf{T}^5_{\mathrm{USB}}$	✓	✓	✓	✓	✓	✓	✓	✓	✓	1

8. Other symmetric key schemes were discussed in [101], [97], [8], [50] but the way in which keys and tags are allocated seems to fit in one of the previous paradigms and thus we do not consider them as separate clases of authentication protocols. A survey on security threats and protocols is available in [80].

Table 2.2: Specifications, advantages and limitations of current proposals for CAN authentication (an updated comparison from the extended version of our conference report [30])

Protocol	Specifications and advantages	Possible Limitations
TESLA-CAN [29]	- efficient use of symmetric primitives with time synchronization - successful and well studied in sensor networks	- fixed authentication delays (usually milliseconds), - time triggered release of keys (potential conflicts with CAN arbitration) - resynchronization can be an issue at very small authentication delays
CAN-Auth [92]	- ID oriented authentication, follows in detail CAN specifications - no authentication delays or time synchronization	- no source authentication (unclear security implications, e.g., third party tools can inject forged frames) - number of keys increases with number of IDs
Voting [83]	- shared symmetric keys between each 2 nodes - each node votes for the authenticity of previous messages	- small number of nodes - time triggered release of authentication tags (nodes need to wait over multiple time slots to get sufficient votes) - nodes are required to be present and vote - disagreements between nodes if packets are lost - each node needs to apply a MAC on his current message, his votes on previous messages, and all messages that he received previously
LiBrA-CAN [30]	- efficient source authentication with dishonest minority - efficient forgery detection with MAC mixing - no authentication delays or time synchronization	- small number of participants - malicious nodes in minority
MaCAN [35]	- simple messsage authentication with shared keys - efficient block cipher based message authentication codes	- attacks found by formal verification (fixable) - potential issues with replay attacks - key distribution procedure unclear
CaCAN [47]	- centralized node automatically discards compromised messages - authentication via one MAC per message - efficient and fully back-ward compatible	- physical removal the master node from the bus leads to complete loss of security - requires more expensive electronics

Chapter 3

Cryptography on wired in-vehicle buses

This chapter presents the protocols that we designed and evaluate in [29], [31], [30]. Our personal contribution consists in the design of the protocols which are presented in the thesis. The experimental results on protocol performance are only briefly presented. We also do include more recent simulation results from [95]. Figure 3.1 tries to give an overview of the proposals so far in a potential chronological order (by publication year), highliting the research done by us at UPT (please refer to Table 2.2 for the exact references to these works). In the following sections we do describe in detail our contributions by the three families of protocols that we employed: TESLA-based schemes [29], one-time signatures based schemes [31] and group keying (LiBrA-CAN) [30].

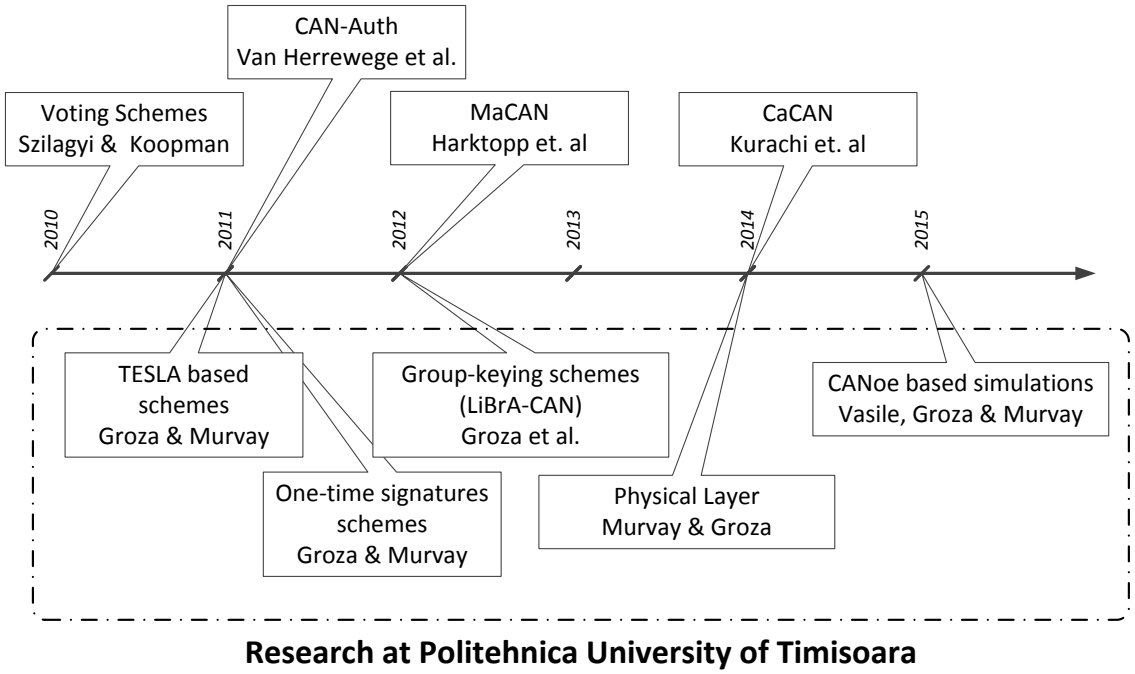


Figure 3.1: Works related to CAN-bus security in potential chronological order (by publication year)

3.1 TESLA-like protocols [1]

In what follows, TESLA based protocols are described based on our results from [29]. Some generic arguments on the choice over this family of protocols as well as some technical arguments on key initialization and time synchronization are first restated based on [29].

The main argument for choosing a TESLA like protocol in our research was that this is the best solution to perform broadcast authentication without secret shared keys or public key primitives. Also, to the best of our knowledge, prior to our work [29] there was no result available to show clear technical limits of using TESLA like protocols on CAN networks.

Some incompatibility between TESLA and CAN appears right from the design of the two, but this is easily surpassed. Indeed, CAN is a message oriented bus while TESLA appears to be source oriented, i.e., it assures that a message originates from a particular sender. We emphasize that there are many practical scenarios in which the source of a particular CAN message does matter and in practice identifiers are frequently uniquely associated to a particular node. Thus the message oriented nature of CAN should not be interpreted in a strict sense, since the source of the message is not always irrelevant. Even for the case of an ID oriented authentication (where authentic messages with the same ID can originate from different nodes) a TESLA like protocol will prove to be more suitable for adding new nodes on the bus since they can authenticate messages via the broadcast scheme without needing to share the secret key for a particular ID.

From an upper view, the design paradigm is the following. Memory, computational speed, bandwidth, initialization time and the synchronization error give bounds on the structure of the chains that we use. This further bounds the authentication delay, i.e, the delay at which authentication keys arrive on the bus, which is crucial to us as messages cannot be authenticated faster than the disclosure delays. To improve on this delay, we design several variants of the protocol that are presented in section 3.

All protocol variants use multiple levels of one-way key chains with the structure suggested in Figure 3.2. The relevant notations with respect to the chain structure are:

- ℓ the number of chain levels,
- $\sigma_i, i = 1..\ell$ the length of the chain on level i,
- $\delta_i, i = 1..\ell$ the disclosure delay on level i,
- ξ the safety margin for releasing authentication packets,
- δ_{norm} the *normalized disclosure delay* a generic delay, to be clarified in the next section.

Informally, in Figure 3.2 bullets depict keys from the key chains and the horizontal black arrows denote that they are derived from a previous key. As usual in such protocols, keys are generated and consumed in a reverse order, thus the time arrow on the bus points in opposite direction to the arrows that generate the keys. Packets arriving on the communication bus are depicted as well, the dotted lines from an element of the chain to the packet denote that the element was used as a key, and for the re-initialization packets in particular one element of the key chain was also used as a message. Packets containing keys are marked by K and commitments, i.e., MAC codes that authenticate forthcoming key chains, are marked by C.

[1] The results presented in this section are based on author's previously published work: Bogdan Groza, Stefan Murvay, *Efficient protocols for secure broadcast in Controller Area Networks*, IEEE Transactions on Industrial Informatics, Volume: 9, Issue: 4, Pages: 2034 - 2042, IEEE, 2013.

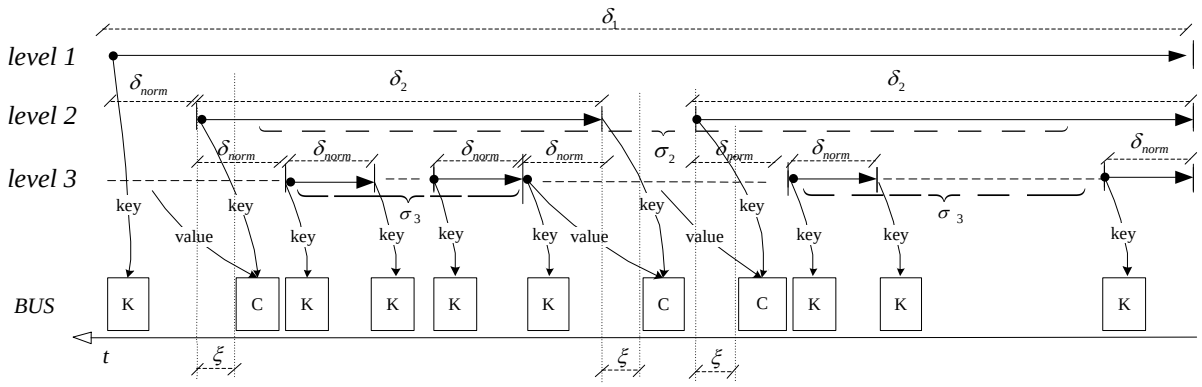


Figure 3.2: Broadcast sequence with normalized time δ_{norm}.

KEY INITIALIZATION. A procedure is necessary in order to set-up the initial broadcast key. In our initial version of this work we claim that this can be build upon standard cryptographic tools, e.g., the PKI. This claim is correct, the problem which remains to be solved is how to make this procedure suitable for the real world, where dozens of ECUs reach the car while being produced by distinct manufacturers. Clearly, this can be only solved by standardization, which is not within reach for us. The same discussion holds for all protocols dedicated to authentication for in-vehicle networks.

TIME SYNCHRONIZATION. Time synchronization is done with respect to a central node, which will play the role of a communication master. As usual, synchronization between two nodes is loose and it requires a handshake and counting the round trip time until it is below a tolerance margin. This is usually achieved in two protocol steps as follows:

$$A \rightarrow B : N_A;$$
$$B \rightarrow A : Sig_B(t_B, N_A).$$

Here N_A denotes a nonce generated by principal A and t_B denotes the current time at principal B when sending its response. Afterwards, the round trip time ϵ_{AB} becomes the synchronization error. If the nonce was sent by A at time t_0 and now A's clock points to t_1 then A knows that the time on B side is in the interval $[t_B + t_1 - t_0,\ t_B + t_1 - t_0 + \epsilon_{AB}]$. However, in our scenario a digital signature costs tens, or hundreds of milliseconds, which will result in an even larger disclosure delay. Because of this, instead of a digital signature we used a message authentication code which is several orders of magnitude faster. In particular, in our experiments, the round-trip-time was in the order of several hundreds micro-seconds as shown in the section dedicated to experimental results.

For rigorous description of the protocol, we need to define the structure of the key chain and the precise timings for the disclosure of the keys. In the original version of the work we used a *timing template* to compute the timings for each level (based on chain lengths and disclosure delays) and a *function template* to generate the keys on each level (based on a one-way function). The *timing template* is an ℓ-tuple of positive integer pairs denoting the chain length and disclosure delay for each particular level, i.e., $\mathbf{T}_\ell = \{(\sigma_1, \delta_1), (\sigma_2, \delta_2), ..., (\sigma_\ell, \delta_\ell)\}$. The *function template* as a ℓ-tuple

of functions that are used to generate the keys on each level, i.e., $\mathbf{F}_\ell = \{\mathsf{F}_1, ..., \mathsf{F}_\ell\}$. The *function template* allows the generation of chains from different levels, with different functions, that will provide good speed-ups for the variants of the protocol that follows.

Now the *indexed key collection* $\mathbb{K}_{\mathbf{T},\mathbf{F}}$ can be defined as a tuple of *time-indexed keys* K_τ, i.e., keys entailed by a vector τ with ℓ elements that defines the exact disclosure time for the key. Given *timing template* $\mathbf{T}_\ell$, *function template* $\mathbf{F}_\ell$ a *time-indexed key* is generated as:

$$\mathsf{K}_{\tau_{left}|\tau_i|\overline{0}} = \mathsf{F}_i(\mathsf{K}_{\tau_{left}|\tau_i+1|\overline{0}}), \forall i \in [1, \ell], \tau_i \in [0, \sigma_i - 1] \tag{3.1}$$

Here, $\mathsf{K}_{\tau_{left}|\sigma_i|\overline{0}}$ is the initialization key for the particular key-chain, computed via a key-derivation process from some random fresh material generated at each initialization as $\mathsf{K}_{\tau_{left}|\sigma_i|\overline{0}} = \mathsf{KD}(\mathsf{K}_{\mathrm{rnd}}, \tau_{left})$, $\mathsf{K}_{\mathrm{rnd}}$ is some fixed random value and KD is a key derivation function. Here τ_{left} is a placeholder for any left part of the index vector τ and the right part, denoted by $\overline{0}$, is always zero.

The previous definition allows the generation of chains on multiple levels with the specified length as suggested in Figure 3.2.

Now we can establish the exact disclosure time for each key. For this we let t_{start} denote the time at which the broadcast was started on the sender side and assuming there are no clock drifts for the sender the exact release time of the keys follows. Given $\mathcal{DT}(\mathsf{K}_\tau, \mathbf{T}_\ell)$ the disclosure time of the *indexed key* K_τ based on *timing template* $\mathbf{T}_\ell$ for a broadcast started at t_{start} is:

$$\mathcal{DT}(\mathsf{K}_\tau, \mathbf{T}_\ell) = t_{start} + \sum_{i=1..\ell} (\delta_i \cdot \tau_i) \tag{3.2}$$

We consider the case of a receiver $\mathcal{R}$ and sender $\mathcal{S}$ with synchronization error $\epsilon_{\mathcal{S},\mathcal{R}}$. To define the security condition that must be met by all packets that contain authentication information, we first need to establish the minimum and maximum time on the $\mathcal{S}$'s side, *estimated* by $\mathcal{R}$ having local clock pointing at $t_\mathcal{R}$. Given synchronization error $\epsilon_{\mathcal{S},\mathcal{R}}$ and $t_\mathcal{S}$ the time value reported by $\mathcal{S}$ on a synchronization performed at t_{sync} with $\mathcal{R}$ it holds:

$$\mathcal{ET}_{\min}(t_\mathcal{R}) = t_\mathcal{R} - t_{sync} + t_\mathcal{S}, \tag{3.3}$$

$$\mathcal{ET}_{\max}(t_\mathcal{R}) = t_\mathcal{R} - t_{sync} + t_\mathcal{S} + \epsilon_{\mathcal{S},\mathcal{R}}. \tag{3.4}$$

Now we define the security condition that must be met by all packets, that is, an authentication packet, i.e., a MAC, computed with K_τ received at time $t_\mathcal{R}$ must be discarded unless:

$$\mathcal{ET}_{\max}(t_\mathcal{R}) \leq \mathcal{DT}(\mathsf{K}_\tau, \mathbf{T}_\ell). \tag{3.5}$$

This condition ensures that an authentication packet is not accepted after the authentication key was already disclosed.

3.1.1 Generic Description of the Protocol

The generic description of the protocol from [29] now follows. We underline that this description does not include particular optimizations that are presented in the section dedicated to practical variants. It works only as a high level description for the forthcoming protocols.

CONSTRUCTION 1. ***(TESLA-CAN Main Scheme)*** *Given* indexed key collection $\mathbb{K}_{\mathbf{T},\mathbf{F}}$ *and two roles called sender and receiver denoted by $\mathcal{S}$ and $\mathcal{R}$ with synchronization error $\epsilon_{\mathcal{S},\mathcal{R}}$, protocol* $\mathsf{Broadcast}_{\mathcal{S},\mathcal{R}}[\mathbb{K}_{\mathbf{T},\mathbf{F}}]$ *is defined by the following actions for the two roles:*

1. $\mathsf{SendKey}(\mathcal{S}, K_\tau, \mathcal{DT})$ *in which sender* $\mathcal{S}$ *sends key* K_τ *at* $\mathcal{DT}(K_\tau)$*,*

2. $\mathsf{SendTag}(\mathcal{S}, \mathsf{MAC}(K_\tau, M), \mathcal{DT})$ *in which sender* $\mathcal{S}$ *sends* $\mathsf{MAC}(K_\tau, M)$ *in any empty time-slot with the condition that* $\mathsf{MAC}(K_\tau, M)$ *is released no latter than* $\mathcal{DT}(K_\tau)$*,*

3. $\mathsf{SendMes}(\mathcal{S}, M)$ *in which sender* $\mathcal{S}$ *sends message* M*, an action which can be performed at any point but is ideally done in the same time interval with the tag* $\mathsf{MAC}(K_\tau, M)$*, i.e., no later than* $\mathcal{DT}(K_\tau)$ *(to simplify the verification step for the receiver we assume that when the key arrives, the message and MAC are already in place),*

4. $\mathsf{RecKey}(\mathcal{R}, K_{\tau_{left}|\tau_i|\bar{0}})$ *on which receiver* $\mathcal{R}$ *receives the key* $K_{\tau_{left}|\tau_i|\bar{0}}$ *which is deemd authentic if and only if* $K_{\tau_{left}|\tau_i|\bar{0}} = \mathcal{F}(K_{\tau_{left}|\tau_i+1|\bar{0}})$ *and* $K_{\tau_{left}|\tau_i+1|\bar{0}}$ *is a previously received authentic key. For each authentic key, the receiver will verify all the corresponing message-tag pairs and deem them authentic in case of successful verification,*

5. $\mathsf{RecTag}(\mathcal{R}, \mathsf{MAC}(K_\tau, M))$ *on which receiver* $\mathcal{R}$ *receives the* $\mathsf{MAC}(K_\tau, M)$ *at* $t_{\mathcal{R}}$ *and discards it if* $\mathcal{ET}_{\max}(t_{\mathcal{R}}) \leq \mathcal{DT}(\mathsf{K}_\tau, \mathbf{T}_\ell)$ *otherwise stores it on a tape,*

6. $\mathsf{RecMes}(\mathcal{R}, M)$ *on which receiver* $\mathcal{R}$ *receives the message* M *and stores it on the tape.*

Here ξ denotes a tolerance margin for the time at which a MAC with a particular key can be sent. Indeed, sending MACs too close to the disclosure time may be useless because the receiver may have to discard them if the security condition cannot be met. Thus ξ must be fixed as initial setup parameter for the protocol. In time interval $[\mathcal{DT}(K_\tau), \mathcal{DT}(K_\tau) + \xi]$ the sender can safely disclose any kind of data packet, but not MACs.

SECURITY ARGUMENT. The security of TESLA-like protocols is well established, a brief security argument is given by us in [29].

3.1.2 Efficiency parameters

In order to discus the possibility of building more practical variants of the main scheme in the section that follows, we first need to fix some synthetic parameters based on [29].

The efficiency of the protocol can be evaluated with respect to memory, CPU and bandwidth. This evaluation has to be done over the entire time horizon of the protocol which can be divided in two parts: initialization time $\mathsf{T}_{\mathsf{init}}$ and runtime $\mathsf{T}_{\mathsf{run}}$. Then, the *generic calibration criteria* for the scheme parameters is the following: for fixed $\mathsf{T}_{\mathsf{run}}$ and δ_{norm} we determine chain structure (lengths and levels) and timings which give the timing template $\mathbf{T}_\ell$, function template $\mathbf{F}_\ell$ and the corresponding key chains $\mathbb{K}_{\mathbf{T},\mathbf{F}}$. Then, we determine bus, CPU and memory loads for comparison. However, bus loads and CPU utilizations, that are going to be defined next, are more relevant only over $\mathsf{T}_{\mathsf{run}}$ as it is natural to expect that during $\mathsf{T}_{\mathsf{init}}$ the initialization can takeover the entire bus and CPU but only for a very short period of time.

NOTATIONS. We will use the following notations: MEM, CPU, BUS and their capacities are depicted in the number of keys that can be stored, computed or sent. For all of these notations, a subscript indicates whether they refer to the initialization stage or the runtime stage. Thus $\mathsf{CPU}_{\mathsf{init}}$ refers to the amount of work during initialization and $\mathsf{CPU}_{\mathsf{run}}$ during runtime. By $\mathsf{MEM}^{\mathsf{total}}$, $\mathsf{CPU}^{\mathsf{total}}$ and $\mathsf{BUS}^{\mathsf{total}}$ we refer to the total available computational power and bus capacity during the entire run-time of the protocol - we use these measures to define CPU and bus loads during runtime. To indicate whether a resource is needed for computing keys or commitments through MAC codes

we use key and com as superscripts. The total number of keys and the number of key chains are denoted as follows:

- $\| \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \|$ denote the total number of keys on level i disclosed during protocol lifetime,
- $\langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle$ the number of key chains from level i.

Obviously we have $\| \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \| = \sigma_i \cdot \langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle$ since the total number of keys is the number of chains multiplied with the chain length. We will use both notations, although it is easy to derive one from the other, in order to make the following relations more intuitive.

PARAMETERS. As disccussed by us in [29], the following overheads on memory, CPU and bus-load hold for the TESLA-CAN protocol:

$$\mathsf{MEM}^{\mathsf{key}}_{\mathsf{init}} = \mathsf{MEM}^{\mathsf{key}}_{\mathsf{run}} = \sum_{i=1,\ell} \sigma_i \cdot m_i \tag{3.6}$$

$$\mathsf{CPU}^{\mathsf{key}}_{\mathsf{init}} = \sum_{i=1,\ell} \sigma_i \cdot c_i \;\; \mathsf{CPU}^{\mathsf{key}}_{\mathsf{run}} = \sum_{i=2,\ell} c_i \cdot (\| \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \| - \sigma_i) \tag{3.7}$$

$$\mathsf{CPU}^{\mathsf{com}}_{\mathsf{init}} = \sum_{i=1,\ell} c_i \;\; \mathsf{CPU}^{\mathsf{com}}_{\mathsf{run}} = \sum_{i=2,\ell} c_i \cdot (\langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle - 1) \tag{3.8}$$

$$\mathsf{BUS}^{\mathsf{key}}_{\mathsf{run}} = \sum_{i=1,\ell} m_i \cdot \| \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \| \tag{3.9}$$

$$\mathsf{BUS}^{\mathsf{com}}_{\mathsf{init}} = \sum_{i=1,\ell} m_i \;\; \mathsf{BUS}^{\mathsf{com}}_{\mathsf{run}} = \sum_{i=2,\ell} m_i \cdot (\langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle - 1) \tag{3.10}$$

Equation 3.6 gives the memory requirements which is the sum of the lengths of the chains. In the case of memory there are no variations during initialization and runtime. More, we do not need additional memory to store commitments on the sender as commitments can be sent as soon as they are computed. Equation 3.7 gives the computational time required for keys at runtime and initialization. During initialization, one chain is computed on each level. At runtime, there are $\langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle$ key-chains on each level, and each of them has to be computed except the first one which was computed during initialization which gives $\mathsf{CPU}^{\mathsf{key}}_{\mathsf{run}} = \sum_{i=2,\ell} c_i \cdot \sigma_i \cdot (\langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle - 1)$. Replacing $\sigma_i \cdot \langle \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \rangle$ with $\| \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \|$ we get the claimed number of keys computed at runtime. In Equation 3.8 commitments are measured: one commitment on each level during initialization, and later for each chain on each level (except for the first one which was committed during initialization) one commitment is needed. Bus requirements for keys during runtime is given in Equation 3.9. All keys from all levels are sent on the bus, while there are no keys (just commitments) sent during initialization. Commitments are given in Equation 3.10. One chain on each level is committed in the initialization, and later at runtime all chains are committed except for the first one, same as in the case of computational requirements.

To complete the view on efficiency, we should also define the CPU and bus loads over the entire lifetime of the protocol. These can be defined as a fraction from the total load. Given $\mathsf{RES} \in \{\mathsf{MEM}, \mathsf{CPU}, \mathsf{BUS}\}$, $\mathsf{state} \in \{\mathsf{run}, \mathsf{init}\}$ we define the protocol overheads as:

$$\mathsf{RES}^{\mathsf{load}}_{\mathsf{state}} = (\mathsf{RES}^{\mathsf{key}}_{\mathsf{state}} + \mathsf{RES}^{\mathsf{com}}_{\mathsf{state}})/\mathsf{RES}^{\mathsf{total}}_{\mathsf{state}}. \tag{3.11}$$

One can add to these the overhead induced by the message authentication codes associated to each data packet that is sent over the bus. This is however application dependent, not protocol dependent as in some applications the size of the data packets can be small, and thus adding a MAC to each data packet will greatly increase the overhead while in other applications it may be the reverse, and data packets can be large and the MAC will not significantly increase the overhead.

3.1.3 Practical versions of the scheme

We now briefly enumerate some practical variations of the scheme as presented in [29]:

1. ***The Multi Master and Single Master Case*** allow each node to be a potential sender. The case of k senders can be easily derived since we can separate the senders by delay δ_{next} (referred as the *next sender delay*). The disclosure timings becomes

$$\mathcal{DT}_k(K_\tau) = \mathcal{DT}(K_\tau) + k \cdot \delta_{next} \tag{3.12}$$

   and the security condition is modified accordingly for the case of k senders. However, allowing more than one sender will result in a bus that is heavily loaded by keys and commitments - thus it is not a preferable option.

2. ***Equidistant Timing (Delayed) Authenticated CAN (ETA-CAN)*** is a solution which assures a uniform bus load. For this, we use a procedure which we call *equidistant timing* by which all keys are disclosed at moments of time separated by equal delays, i.e.,

$$\delta_\ell = \delta_{\mathsf{norm}} = \mathsf{T}_{\mathsf{run}} / \sum_{i=1,\ell} \parallel \mathbb{K}_{\mathbf{T},\mathbf{F}}[i] \parallel = \mathsf{T}_{\mathsf{run}} / \prod_{i=1,\ell} (\sigma_i + 1) - 1, \tag{3.13}$$

$$\delta_i = \delta_{\mathsf{norm}} \cdot \prod_{j=i+1,\ell} (\sigma_j + 1), 1 \leq i < \ell. \tag{3.14}$$

3. ***Balanced Equidistant Timing delayed Authenticated CAN (BETA-CAN)*** uses chains of equal sizes on all levels. Since the entire run-time of the protocol is $\mathsf{T}_{\mathsf{run}} = \delta_{\mathsf{norm}} \cdot [(\sigma + 1)^{\ell^{\mathsf{BETA}}} - 1]$ the number of levels follows as:

$$\ell^{\mathsf{BETA}} = \lceil \log_{\sigma+1} \left(\frac{\mathsf{T}_{\mathsf{run}}}{\delta_{\mathsf{norm}}} + 1 \right) \rceil. \tag{3.15}$$

   The disclosure delay of the last level is δ_{norm} while for the upper levels the delay can be computed as:

$$\delta_i^{\mathsf{BETA}} = \delta_{\mathsf{norm}} \cdot (\sigma + 1)^{\ell^{\mathsf{BETA}} - i}. \tag{3.16}$$

4. ***Ad hoc secure Balanced levels ETA-CAN (Ad-BETA-CAN)*** increases performance by the use of reduced versions of hash functions, that is by truncating their output. This is natural choice but the precise value of the security parameters needs to be carefully established in practice.

5. ***Ad hoc secure Greedy last level ETA-CAN (Ad-GETA-CAN)*** uses a greedy strategy in order to minimize memory overheads. Given δ_{norm} and $\mathsf{T}_{\mathsf{init}}$ we first select the less intensive function that is ad hoc secure with respect to δ_{norm}. Then we use the entire $\mathsf{T}_{\mathsf{init}}$ time to compute a chain from the last level subject only to memory constraints. Then we set to 1 the length of each chain from level 1 to $\ell - 1$. In this way we minimize the memory and computational time for the upper layers. Since the number of upper layers is maximum, due to the reduced chain length, this also increases initialization overhead on the bus. $\mathsf{T}_{\mathsf{init}}$ should be only slightly overloaded since the initialization time is minimum when the number of levels is maximum. If memory is also exhausted by the first layer, we cut down as many elements as are needed to fit the upper layers. The parameters of the scheme are computed by the relations given below. Having $\mathsf{T}_{\mathsf{run}} = \delta_{\mathsf{norm}} \cdot [2^{\ell^{\mathsf{GETA}}-1} \cdot (\sigma + 1) - 1]$ the number of levels and the disclosure delay are:

$$\ell^{\mathsf{GETA}} = \left\lceil \log_2 \frac{\mathsf{T}_{\mathsf{run}} + \delta_{\mathsf{norm}}}{\delta_{\mathsf{norm}} \cdot (\sigma + 1)} \right\rceil + 1, \tag{3.17}$$

$$\delta_i^{\mathsf{GETA}} = \delta_{\mathsf{norm}} \cdot 2^{\ell^{\mathsf{GETA}}-i-1} \cdot (\sigma + 1). \tag{3.18}$$

For more details, the full version of our work [29] can be consulted.

3.1.4 Synthetic efficiency evaluation

We give only a brief account of the efficiency analysis in [29]. Having these defined the performance indicators for memory, CPU and bus can be derived. These indicators are summarized for all variants of the schemes in Table 3.1.

Figure 3.3 shows the influence of the chain lengths and levels on various parameters. Plots are taken for a time range $\mathsf{T}_{\mathsf{run}} = 24$ hours while the bus speed is approximated to about 6000 packets per second. In the case of variations with chain lengths, plots (i) and (ii) are given for three and four levels of key chains. We note that the delays drop rapidly by increasing the number of levels in plot (i), but in the same manner the overhead increases (ii) (at 100% the bus is locked and communication halted). Plot (iii) shows the variation of memory requirements, which is the same as the initialization time, and plot (iv) of commitments with the number of chain levels. The same drop in the memory requirements and increase in the number of commitment packets can be seen. For plots (iii) and (iv) the delay is fixed to 5 ms.

Figure 3.4 shows the generic difference between the $BETA$ and $GETA$ schemes assuming fixed delay and variable length or the reverse. $BETA$ gives a lighter bus due to fewer chain commitments, however, $GETA$ is much better in terms of memory requirements, also a relevant constraint of our environment.

3.2 One-time signature based schemes [2]

Our work in [31] was focused on the use of one-time signatures to assure broadcast authentication. This was a natural approach since one-time signatures were specifically designed to sign messages by the use of simple trapdoor-less one-way functions rather than by more expensive public key primitives. We do of course not intend to achieve non-repudiation, as we just want to authenticate

[2] The results presented in this section are based on author's previously published work: Bogdan Groza, Stefan Murvay, *Secure Broadcast with One-time Sigantures in Controller Area Networks*. Proceedings of International Conference on Availability, Reliability and Security (ARES'11), IEEE Comp. Soc., 2011.

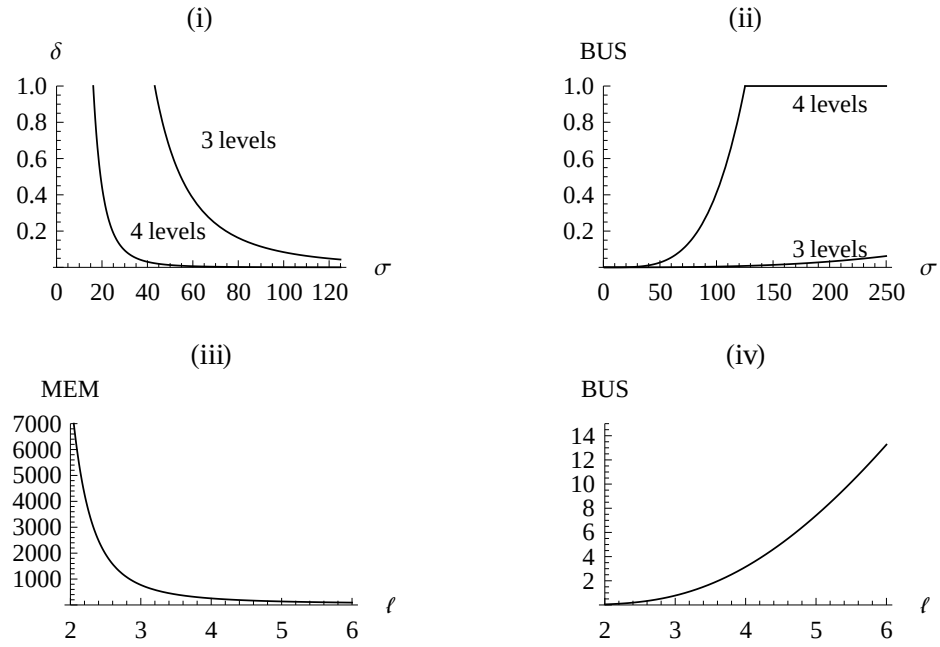


Figure 3.3: Variation of overheads and delays with length or levels: (i) disclosure delay , (ii) overhead caused by keys and commitments on the bus, (iii) keys stored in memory and (iv) commitments on the bus (per second). (as depicted in [29])

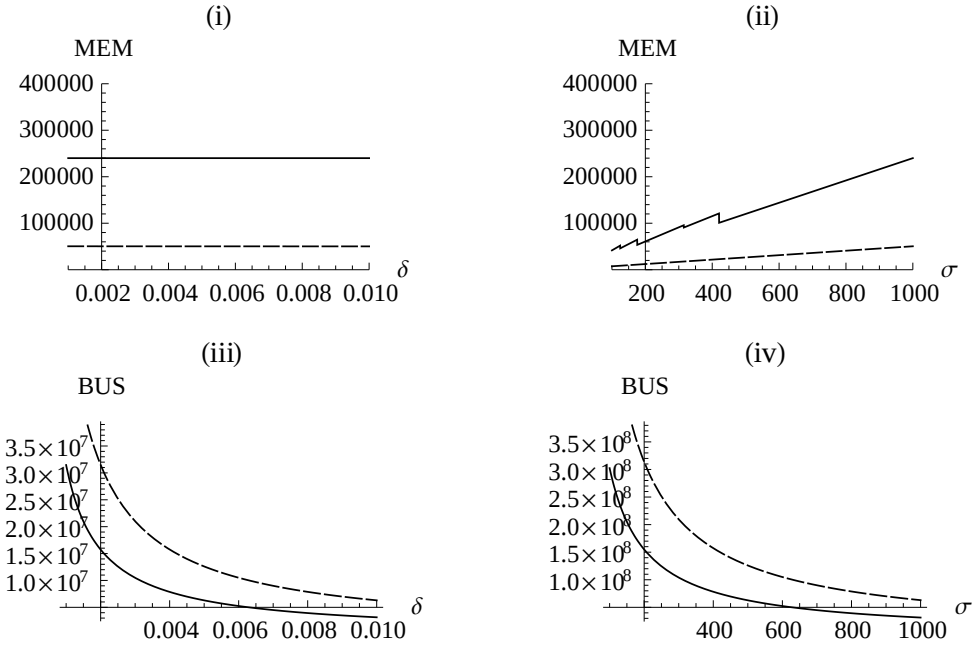


Figure 3.4: Comparison between $BETA$ (continuous line) and $GETA$ (dotted line) schemes at MEM and BUS requirements for: fixed $\sigma = 1000$ and variable δ_{norm} in (i) and (iii), variable δ_{norm} and fixed $\sigma = 1000$ in (ii) and (iv). (as depicted in [29])

Table 3.1: Some overheads at initialization and run-time (as established in [29])

	BETA-CAN	Ad-BETA-CAN
$\mathsf{MEM}^{\star}_{\mathsf{key}}$	$\ell \cdot \sigma \cdot m$	$\sigma \cdot \sum_{i=1,\ell} m_i$
$\mathsf{CPU}^{\mathsf{init}}_{\mathsf{key}}$	$\ell \cdot \sigma \cdot c$	$\sigma \cdot \sum_{i=1,\ell} c_i$
$\mathsf{CPU}^{\mathsf{init}}_{\mathsf{com}}$	$\ell \cdot c$	$c_0 + \sum_{i=1,\ell-1} c_i$
$\mathsf{BUS}^{\mathsf{init}}_{\mathsf{com}}$	$\ell \cdot m$	$m_0 + \sum_{i=1,\ell-1} m_i$
$\mathsf{CPU}^{\mathsf{run}}_{\mathsf{key}}$	$[(\sigma+1)^{\ell} - \sigma \cdot \ell - 1] \cdot c$	$\sigma \cdot \sum_{i=2,\ell}[(\sigma+1)^{i-1} - 1] \cdot c_i$
$\mathsf{BUS}^{\mathsf{run}}_{\mathsf{key}}$	$[(\sigma+1)^{\ell} - 1] \cdot m$	$\sigma \cdot \sum_{i=1,\ell}(\sigma+1)^{i-1} \cdot m_i$
$\mathsf{CPU}^{\mathsf{run}}_{\mathsf{com}}$	$[\frac{(\sigma+1)^{\ell}-1}{\sigma} - \ell] \cdot c$	$\sum_{i=2,\ell}[(\sigma+1)^{i-1} - 1] \cdot c_i$
$\mathsf{BUS}^{\mathsf{run}}_{\mathsf{com}}$	$[\frac{(\sigma+1)^{\ell}-1}{\sigma} - \ell] \cdot m$	$\sum_{i=2,\ell}[(\sigma+1)^{i-1} - 1] \cdot m_i$

the messages, a reason for which we can drop on Merkle trees to re-initialize the keys. The following subsection presents the signature schemes, then we proceed to the protocol design and finally to some efficiency results. The results prove that these scheme are not the best alternative, they may perform better than standard public-key cryptography but there is still much before competing with the standard MAC-based schemes. From the following exposition we skip some of the arguments on security which can be retrieved from the original version of the work [31].

3.2.1 The employed signature schemes: a Merkle based scheme and HORS

We first present the one-time signature schemes as considered by us in [31]. The generic principle behind both of the one-time signature schemes is to apply a simple on-way function, e.g., a hash function, over some input that plays the role of a secret key and use the output as public key. However if bits are signed individually this results in an inefficient scheme, not necessarily due to the number of hash computations since these are cheap, but mainly due to the size of the signature itself, e.g., in the worst case one hash for each bit. For this purpose, several improvements were proposed in the literature. The enhanced Merkle signature and HORS [69] that we discuss next employ the one-way chains in two highly distinctive fashions, a reason for which we choose to evaluate both of them in our CAN broadcast scenario.

Enhanced Merkle Signature Scheme (EMS). Given one-way function f, signature scheme EMS is a triplet of polynomial time algorithms $Gen, Sign, Ver$ where:

1. Gen is a probabilistic algorithm that takes as input the security level k along with two integers λ, μ and outputs the public-private key pair pk, pv, i.e., $pb = \{(f^{\lambda}(u_{\mu}), f^{\lambda}(v_{\mu}))$,..., $(f^{\lambda}(u_2), f^{\lambda}(v_2)), (f^{\lambda}(u_1), f^{\lambda}(v_1))\}$, $pv = \{(u_{\mu}, v_{\mu}), ..., (u_1, v_1)\} \leftarrow Gen(1^k, \lambda, \mu)$ (here all u_i, v_i are random values of k bits each),

2. $Sign$ is a deterministic algorithm that takes as input the private key pv, a message m of $\lfloor log_2(\lambda) \rfloor \cdot \mu$ bits which can be written as $m = m_{\mu}...m_2 m_1$ (where each m_i has $\lfloor log_2(\lambda) \rfloor$ bits), and outputs a signature s, i.e., $s = \{(f^{\lambda - m_{\mu}}(u_{\mu}), f^{m_{\mu}}(v_{\mu})), ..., (f^{\lambda - m_2}(u_2), f^{m_2}(v_2)), (f^{\lambda - m_1}(u_1), f^{m_1}(v_1))\}$,

3. Ver is a deterministic algorithm that takes as input the signature and the public key and outputs message $m = m_\mu ... m_2 m_1$ if and only if $\forall i = 1..\mu, f^{\lambda - m_i}(s_i) = f^\lambda(u_i), f^{m_i}(s_i) = f^\lambda(v_i)$ or $\perp$ otherwise.

HORS Signature Scheme [69]. Given one-way function f, signature scheme HORS is a triplet of polynomial time algorithms $Gen, Sign, Ver$ where:

1. Gen is a probabilistic algorithm that takes as input the security parameters l, k and integers λ, μ then generates λ random k-bit values $s_1, s_2, ..., s_\lambda$ then computes $v_i = f(s_i)$ and outputs the public-private key pair pk, pv, i.e., $pb = \{\mu, f(s_1)$,..., $f(s_\lambda)\}$, $pv = (k, s_1, ..., v_\lambda) \leftarrow Gen(1^k, l, \lambda)$,

2. $Sign$ is a deterministic algorithm that takes as input the private key pv and message m then computes $h = hash(m)$ and splits h into k substrings $h_1, ..., h_\mu$ each of $\log_2(t)$ bits and outputs $s = (s_{h_1}, ..., s_{h_\mu})$ (where each h_i is interpreted as an integer index),

3. Ver is a deterministic algorithm that takes as input the signature s, the public key pb and message m then outputs 1 if and only if $f(s'_i) = v_i$ for each i extracted as integer index from $h(m)$.

3.2.2 The broadcast protocol

We now describe the broadcast protocol that was studied by us in [31]. For each of the signature schemes we use a broadcast protocol that relies on one-way key chains. In the case of the EMS signature, the key chain is used to commit future public keys, while in the case of the HORS signature each element of the key chain forms a public key for the signature (this happens in a similar fashion to the BiBa protocol from Perrig [63]).

Time synchronization is loose and is done with synchronization error $\epsilon_{\mathcal{R},\mathcal{S}}$ which is the round-trip time of a handshake between the receiver and the sender. How to achieve this synchronization was already discussed in the previous section dedicated to TESLA-like protocols.

CONSTRUCTION 2. ***(Broadcast with EMS)*** *Given signature scheme* EMS *and the roles sender $\mathcal{S}$ and receiver $\mathcal{R}$ we define protocol* Broadcast-EMS$_{\mathcal{S},\mathcal{R}}[\lambda, \mu, \delta]$ *as the following actions performed by $\mathcal{S}$:*

1. Initialization: $\mathcal{S}$ generates a key chain by using a random $\mathbb{k}_0$ and computing $\mathbb{k}_n = f(\mathbb{k}_{n-1}), \forall i = 1..n$, then he commits the tip of its top level-chain, i.e., $\mathbb{k}_n$, the disclosure delay δ and the public key obtained by running $Gen(1^k, \lambda, \mu)$,

2. Commitment: $\mathcal{S}$ sends at any point in time interval $[t_{start} + i \cdot \delta, t_{start} + (i+1) \cdot \delta - \xi]$ (here ξ is a tolerance value to prevent the sender to commit a MAC *to close to the disclosure point which will increase the chance for a receiver to drop the packet) a fresh public key pb generated by using $Gen(1^k, \lambda, \mu)$ and a* MAC *computed with $\mathbb{k}_{i+1}$ on it, i.e,* MAC$_{\mathbb{k}_{i+1}}(pb)$,

3. Key Disclosure*: $\mathcal{S}$ sends at time $t_{start} + i \cdot \delta$ the corresponding key from the key chain, i.e., $\mathbb{k}_i$,*

4. Authentic Broadcast: $\mathcal{S}$ sends at any time in $[t_{start} + i \cdot \delta, t_{start} + (i+1) \cdot \delta - \xi]$ message m as a signature with message recovery $s = Sign(m, pv_{last})$ (here pv_{last} is the most recently generated private key);

and $\mathcal{R}$ respectively:

1. Initialization: $\mathcal{R}$ receives the initialization information of the sender, i.e., $\mathbb{k}_n$, the disclosure interval δ and the public key pk,

2. Time Synchronization: $\mathcal{R}$ performs a loose time synchronization with $\mathcal{S}$, such that the synchronization error $\epsilon_{\mathcal{R},\mathcal{S}} << \delta$,

3. Receive Keys and Commitments: $\mathcal{R}$ receives $\mathbb{k}_i$ and checks if $f(\mathbb{k}_i) = \mathbb{k}_{i-1}$ and discards it otherwise. Any MAC *computed with $\mathbb{k}_i$ that is received after $\mathcal{T}_{\mathcal{S},\mathcal{R}}(i \cdot \delta)$ is discarded. Any public key for which there exists a valid* MAC *and key $\mathbb{k}$ that can verify it is deemed authentic,*

4. Message Verification: $\mathcal{R}$ runs $Ver(pk_i, sig_i)$ for any valid public key and deems authentic any output different from $\perp$.

CONSTRUCTION 3. ***(Broadcast with HORS)****. Given signature scheme* HORS *and the roles sender $\mathcal{S}$ and receiver $\mathcal{R}$ we define protocol* Broadcast-HORS$_{\mathcal{S},\mathcal{R}}[\lambda, \mu, \delta]$ *as the set of following actions performed by $\mathcal{S}$:*

1. Initialization*: $\mathcal{S}$ generates a key chain starting from random $\mathbb{k}_0, ..., \mathbb{k}_\lambda$ and computing $\mathbb{k}_j^i = f(\mathbb{k}_i^{j-1}), \forall i = 1..\lambda, j = 1..\mu$, then he commits the tip each chain, i.e., $\mathbb{k}_i^\mu$,*

2. Authentic Broadcast*: $\mathcal{S}$ sends at any time in $[t_{start} + i \cdot \delta, t_{start} + (i+1) \cdot \delta - \xi]$ message m along with its signature computed with* HORS *having as secret key input the keys from the current disclosure interval $\mathbb{k}_0^i, \mathbb{k}_1^i, ..., \mathbb{k}_\lambda^i$ (the number of messages signed in each time interval depends on the security level and signature parameters);*

3. Key Disclosure*: $\mathcal{S}$ sends at time $t_{start} + i \cdot \delta$ all the keys from the current interval that were not disclosed as HORS signature (to save some bandwidth, sending these keys can be skipped since the receivers can validate future signatures with previously received keys, but note that this will increase verification time on receivers)*

and $\mathcal{R}$ respectively:

1. Initialization*: $\mathcal{R}$ receives the initialization information of the sender, i.e., $\mathbb{k}_i^\mu, \forall i = 1..\lambda$ and the disclosure interval δ,*

2. Time Synchronization*: $\mathcal{R}$ performs a loose time synchronization with $\mathcal{S}$, such that the synchronization error $\epsilon_{\mathcal{R},\mathcal{S}} << \delta$,*

3. Receive Keys and Commitments*: $\mathcal{R}$ receives keys $\mathbb{k}_i^j$ and checks if $f(\mathbb{k}_i^{j-1}) = \mathbb{k}_i^j$ and discards it otherwise.*

4. Message Verification*: $\mathcal{R}$ runs $Ver(pk_i, sig_i)$ for any signature that is received in the correct time interval and deems authentic any input that is correctly verified.*

3.2.3 Efficiency analysis and comparison

For the complete efficiency analysis we do refer the reader to our original report in [31], here we only give a short overview of the results. In [31] we analysed various trade-offs that can be achieved with the enhanced Merkle signature then we compare it to RSA signatures and finally to HORS. The

main conclusion was that in general HORS would be more efficient in terms of verification speed and bus load while it is less efficient in terms of memory requirements. A theoretical estimation for the EMS signature example on fault tolerant CAN at 128kbps, led to the conclusion that number of signed bits will not exceed 1.2kbps even if a hash computation does not exceed $100\mu s$. For a bus speed at 1 Mbps, i.e., high speed CAN, the number of signed bits can get up to 2.5 kbps. Both these values seem to be too low for practical needs. The original version of our work [31] reports verification dealys in the order of a dozen milliseconds. For this reason, the scheme cannot be considered more efficient than TESLA (previously presented) or with LiBrA-CAN (presented in the following section).

3.3 Group keying (LiBrA-CAN) [3]

LiBrA-CAN was initially published by us as a conference report in [30]. However, since then we made significant progress in improving our work, e.g., adding more details, new protocol versions and experimental results, and an extended version of our report on LiBrA-CAN can be found on author's website which is not yet published but still under submission to a journal. The details that are included in this section are mostly based on the extended report and not on the short conference version of our paper [30].

LiBrA-CAN is based on two paradigms: key splitting and MAC mixing, the later procedure is optional and is intended to increase security by allowing each node to detect a potential forgery. In addition to these, authentication can be achieved in a progressive manner by revealing only a few bits of the MAC in the case of the smaller standard CAN frames, while for the larger CAN-FD frames we take advantage of the extended data field to increase the security level and reduce the busload. Key splitting allows a higher entropy for each mixed MAC that is sent at the cost of losing some security for groups that contain more malicious nodes. An adversarial majority will be required to break the protocol, while if there are fewer adversarial nodes, the security level is drastically increased. Consequently, this appears to give a flexible and efficient trade-off. Note that in contrast to the scheme of Szilagyi and Koopman which requires the nodes to be present and vote, LiBrA benefits from a majority of non-corrupted nodes, but does not require their presence to vote (it is just their keys that need to be safe). This procedure is not new, similar techniques were proposed in the past in the context of broadcast security. We could trace this back up to the work of Fiat and Naor [21], but there is a large amount of papers on this subject. The work of Canetti et al. [14] provides efficient constructions based on the same principles. However, the constraints of our application in CAN networks are entirely different from related work where this procedure was suggested or used in scenarios such as sensor networks [16], pay-TV [61], etc. The main idea behind such schemes is that groups of l corrupted receivers cannot learn the secret (in settings with $n > l$ users). One interesting feature of such protocols, which we consider relevant for the setting here, is the ability to trace corrupted keys [61]. While this feature is not directly exploited in our protocol, it can be used in our setting as well to detect malicious nodes (roughly, a corrupted receiver has some chances in forging a MAC but the probability that his forgery is detected increases exponentially with the number of forgeries). In addition to this, we exhibit a distinct contribution in the construction of Linearly Mixed MACs which allows us to amalgamate

[3] The results presented in this section are based on author's previously published work: Bogdan Groza, Stefan Murvay, Anthony van Herrewege, Ingrid Verbauwhede, *LiBrA-CAN: a Lightweight Broadcast Authentication protocol for Controller Area Networks*, Proc. 11th Intl. Conf. on Cryptology and Network Security (CANS'12), Springer-Verlag, LNCS vol. 7712, 2012.

several authentication tags in a single tag via a system of linear equations. This construction has the advantage that if one of the MACs is wrong then this will affect all other MACs and thus the mixed MAC will fail to verify on any of the multiple keys. This increases the chance of a forgery being detected and ultimately it increases the reliability in case that benign nodes are in possession of a wrong key. To the best of our knowledge this procedure is new. The closest work that we could find are the multi-verifier signatures proposed by Roeder et al. [72]. In their work, linear systems of equations are used as well upon message authentication codes but the security properties and goals of their construction are different. For our construction we require that the mixed MACs are strongly non-malleable, a property which appears to be entirely different. Another assumption behind our design is that the number of nodes that are connected to the same bus is not large. While indeed ECUs inside cars come from different manufacturers which may or may not be trustworthy, we believe that suspicious ECUs should be limited in number, since the potential insertion of a trapdoor in some component will discredit the public image of the manufacturer too much and there appears to be little or no benefit for this. Moreover, ECUs that come from the same manufacturer and are trustworthy with each other may potentially use the same shared key (randomly generated at runtime for each sub-network in which they are plugged). In this way the number of actual keys needed to assure broadcast security should be smaller than it appears to be at first sight. In our design we try to take advantage of this assumption, and our approach is especially efficient in the case when compromised nodes form only a minority. The idea of malicious nodes in minority is also supported by research subsequent to our initial conference report on LiBrA-CAN [30]. Most of the research works suggest that the attack comes either from a malicious device inserted on the bus or from compromising the software inside a single ECU [47] , [101], [97]. Grouping nodes under the same secret shared key is also proposed in [35] where groups are formed on the trust-level on the node.

It is also to the advantage of this proposal that a new CAN standard that supports flexible data rates was released: CAN-FD [71]. The extended version of our report [30] includes experimental results by simulation with the industry standard CANoe tool from Vector (www.vector.com).

We begin with a brief overview of the application setting and assumptions. Then we outline the main authentication scheme and discuss some variations or improvements to it.

3.3.1 Assumptions and goals

We do assume the usual presence of a Dolev-Yao adversary that has full control over the communication channel. That is, he can eavesdrop, modify and send messages at his will. Of course, the goal of our scheme is to assure an authentic channel, i.e., to prevent messages that originate from the adversary to be accepted by the honest principals. This is achieved by authentication tags added to each message or separately sent (according to the protocol variant). Further, the security and efficiency gain stem from the way in which we share keys between the nodes (avoiding the simple but less efficient pair-wise sharing) and the way in which we build the authentication tags (while a simple concatenation of the authentication tags can be also employed in our scheme, we view this rather as a basic approach and propose the more elegant linear mixing).

FRAME STRUCTURE. For the case of standard CAN frames that carry at most 8 bytes of data and are thus unable to handle both the data and authentication tag, as well as for some variations of the main scheme, we separate between message frames and authentication frames. Larger data blocks or authentication tags (exceeding 8 bytes) can be split across multiple frames with the same ID field and counter. On the other side, with CAN-FD which allows up to 64 bytes of data to be

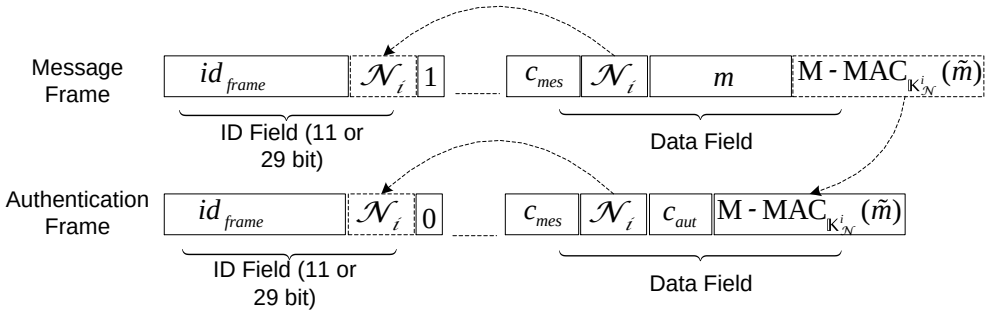


Figure 3.5: Data frames and authentication frames

carried by one frame, it is advantageous to embed the authentication tag in the message frame and take benefit of the increased data rate that follows the arbitration procedure. This also reduces the authentication delay and allows immediate verification.

In Figure 3.5 we suggest the structure of the frame for the case when the authentication tag, i.e., $\mathsf{M\text{-}MAC}_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m})$, is embedded in the message frame and we also outline the case when it is sent as a separate authentication frame (dashed arrow). In both cases the frame structure consists in the identifier of the message id_{frame} which is the usual CAN ID, the identifier of the source node $\mathcal{N}_i$, a message counter c_{mes}, the message itself m and the authentication tag $\mathsf{M\text{-}MAC}_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m})$. Supplementary, in the case of authentication frames, a new counter c_{aut} specifies the number of the authentication frame (intended for protocol variants where there is more than 1 authentication frame for a message). The last bit of the identifier field specifies whether a frame carries an authentication tag or message (1 vs. 0). In the experimental results from [30] we used separate authentication frames with CAN capable boards, while in the CAN-FD simulation from CANoe the data frames carries the authentication tag as well (allowing immediate authentication). Further adjustments can be done. For example, since the data field is quite short, the node identifier (which denotes the source of the authentication frame) can be moved in the ID field (suggested by a dashed arrow in Figure 3.5).

MESSAGE FRESHNESS. This issue brings a more complex discussion for which we refer to the extended version of our report [30]. The reason for not including it here is that it is a more complex issue which is rather bypassed than solved by the research papers. In the absence of a practical deployment for a real-world in-vehicle network it is hard to claim that it is solved. Briefly, related work [35] claims to solve it by timestamps, but these require synchronization and due to the synchornization errors it is likely that replay attacks are possible within the timeframe of the synchronization error. For this reason we do work with counters as explicitly shown in the frame structure. We do howveevr admit that these will eventually overflow, a reason for which it is likely they should be mixed with timestamps in practical deployments (a lenghtier discussion is not within the scope of the current exposition).

MIXED MESSAGE AUTHENTICATION CODES. In the main scheme we make use of Mixed Message Authentication Codes (M-MAC) which amalgamate more MACs into one. The M-MAC uses an array of keys to build a tag which is verifiable by any of the keys. The easiest way to build an M-MAC is simply by concatenating multiple tags, such a construction is fine for our protocol

and can be safely embodied in the main scheme (however, we do further improve on this trivial construction). The generic M-MAC construction is described next.

CONSTRUCTION 4. ***(Mixed Message Authentication Code)*** *A mixed message authentication code* M-MAC *is a tuple* $(\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ *of probabilistic polynomial-time algorithms such that:*

1. $\mathbb{K} \leftarrow \mathsf{Gen}(1^\ell, s)$ *is the key generation algorithm which takes as input the security parameter* ℓ *and set size* s *then outputs a key set* $\mathbb{K} = \{\Bbbk_1, ..., \Bbbk_s\}$ *of* s *keys,*

2. $\tau \leftarrow \mathsf{Tag}(\mathbb{K}, \mathbb{M})$ *is the* MAC *generation algorithm which takes as input the key set* $\mathbb{K}$ *and message tuple* $\mathbb{M} = (m_1, ..., m_s)$ *where each* $m_i \in \{0,1\}^*$ *then outputs a tag* τ *(whenever needed, to avoid ambiguities on the message and key, we use the notation* $\mathsf{M\text{-}MAC}_{\mathbb{K}}(\mathbb{M})$ *to depict this tag),*

3. $v \leftarrow \mathsf{Ver}(\Bbbk_i, m_i, \tau)$ *is the verification algorithm which takes as input a key* $\Bbbk_i \in \mathbb{K}$*, a message* $m_i \in \{0,1\}^*$ *and a tag* τ *and outputs a bit* v *which is* 1 *if and only if the tag is valid with respect to the key* $\Bbbk_i$ *and message* m_i*, otherwise the bit* v *is* 0*. For correctness we require that if* $\Bbbk_i \in \mathbb{K}$ *and* $m_i \in \mathbb{M}$ *then* $1 \leftarrow \mathsf{Ver}(\Bbbk_i, m_i, \mathsf{M\text{-}MAC}_{\mathbb{K}}(\mathbb{M}))$.

A practical instantiation for such a construction is provided in what follows with the LM-MAC and then we provide a simplication of it to gain more efficiency.

CONSTRUCTION 5. ***(Linearly Mixed MAC)*** *We define the* LM-MAC *as the tuple of probabilistic polynomial-time algorithms* $(\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ *that work as follow:*

1. $\mathbb{K} \leftarrow \mathsf{Gen}(1^\ell, s)$ *is the key generation algorithm which flips coins and returns a key set* $\mathbb{K} = \{\Bbbk_1, ..., \Bbbk_s\}$ *where each key has* ℓ *bits (*ℓ *is the security parameter of the scheme),*

2. $\tau \leftarrow \mathsf{Tag}(\mathbb{K}, \mathbb{M})$ *is the* MAC *generation algorithm which returns a tag* $\tau = \{x_1, x_2, ..., x_s\}$ *where each* x_i *is the solution of the following linear system in* $GF(2^b)$*:*

$$\begin{cases} \mathsf{KD}_1(\Bbbk_1, m_1) \cdot x_1 + ... + \mathsf{KD}_s(\Bbbk_1, m_1) \cdot x_s \equiv \mathsf{MAC}_{\Bbbk_1}(m_1) \\ \mathsf{KD}_1(\Bbbk_2, m_2) \cdot x_1 + ... + \mathsf{KD}_s(\Bbbk_2, m_2) \cdot x_s \equiv \mathsf{MAC}_{\Bbbk_2}(m_2) \\ ... \\ \mathsf{KD}_1(\Bbbk_s, m_s) \cdot x_1 + ... + \mathsf{KD}_s(\Bbbk_s, m_s) \cdot x_s \equiv \mathsf{MAC}_{\Bbbk_s}(m_s) \end{cases}$$

Here b *is polynomial in the security parameter* ℓ *and* KD *stands for a key derivation process. If such a solution does not exist, then the* M-MAC *algorithm fails and returns* $\perp$*.*

3. $v \leftarrow \mathsf{Ver}(\Bbbk_i, m_i, \tau)$ *is the verification algorithm which returns* 1 *if and only if having* $\tau = \{x_1, x_2, ..., x_s\}$ *it holds* $\mathsf{MAC}_{\Bbbk_i}(m_i) \equiv \mathsf{KD}_1(\Bbbk_i, m_i) \cdot x_1 + \mathsf{KD}_2(\Bbbk_i, m_i) \cdot x_2 + ... + \mathsf{KD}_s(\Bbbk_i, m_i) \cdot x_s$*. Otherwise it returns* 0*.*

CONSTRUCTION 6. ***(Simplified Linearly Mixed Message Authentication Code)*** *We define the* SLM-MAC *in the same manner as the* LM-MAC *except for the fact that in the generation and verification algorithms the message is not used by the key derivation process, i.e.,* $\mathsf{KD}_i(\Bbbk_j, m_j)$ *is replaced by* $\mathsf{KD}_i(\Bbbk_j), \forall i, j \in 1..s$*.*

SECURITY PROPERTIES AND PROOFS FOR M-MAC. The first security property which we require for an M-MAC is *unforgeability* which is a standard property for any MAC code. We also require a new property which we call *strong non-malleability* and which lets any verifier detect

whenever the adversary had tampered with any part of the M-MAC. We show that both these properties are achievable by the proposed constructions in proof that can be found in the extended version of our work [30].

3.3.2 LiBrA-CAN - the main scheme

We provide here the main scheme of LiBrA-CAN as presented in the extended version of our conference report [30]. In our initial conference report [30] we defined the main authentication scheme around a master oriented communication. This was justified by the fact that due to the limited size of a standard CAN frame [70] one frame would not be enough to carry both the message and the authentication tag. Consequently, using a master node with higher computational power to continue the authentication seemed like a correct practical approach, justified also by the results of the experimental section. However, the master oriented communication may somewhat conflict with CAN specifications (which clearly specify that CAN is a multi-master bus) and it also results in more overhead by sending multiple authentication frames (notably, CAN frames have about 50% overhead). Fortunately, as we worked on the protocol, the new CAN standard with flexible data rates CAN-FD was released [71] and this allows us to place all the authentication information in a single frame, reducing the overhead and making it possible to have a cleaner, crisper protocol specification (we also include results from a real-time simulation of the main scheme on CAN-FD).

NOTATION. To avoid unnecessary formalism that would not impact security we make some simplifications. Whenever the authentication tag does not fit in a single frame we assume that it is sent over separate authentication frames each of them having the proper counter c_{aut} (we do not explicitly use c_{aut} in the description of the schemes since this will only overload the notations). For the same reason, we use the notation $\widetilde{m}$ to denote the message that is already augmented by the counter c_{mes}, identifier id_{frame} and node identity $\mathcal{N}_i$ (the node identity $\mathcal{N}_i$ can be eventually skipped if it is embedded in the ID field of the frame). Since, with one exception, all versions of the protocol authenticate the same message to all nodes (rather than authenticate a tuple of messages), we replace $\mathbb{M}$ with the augmented message $\widetilde{m}$ and we write $\mathsf{M\text{-}MAC}_{\mathbb{K}}(\widetilde{m})$. Obviously in this case the M-MAC receives as input a message tuple of s identical messages $\widetilde{m}$, i.e., $\mathbb{M} = \{\underbrace{\widetilde{m}, \widetilde{m}, ..., \widetilde{m}}_{s-times}\}$.

The key allocation procedure distributes the keys to the n nodes by placing them in groups of size g. Figure 3.8 provides an example of key-sharing for the master oriented version of the scheme, for the main scheme every sender will be placed in the role of the master. Figure 3.6 is suggestive for what we will later introduce as the LiBrA-CAN main scheme.

KEY SHARING PROCEDURE. There are numerous key-sharing procedures proposed in the literature based on the two well-known and explored alternatives: a secure symmetric-key server, e.g, employed in related work from [35] and [101], or the public key infrastructure (PKI), e.g., suggested in related work from [100]. However, the biggest challenge in the context of in-vehicle security is not in writing some key-exchange protocol but in finding one that can be integrated in real-world deployments where components from a single car come from many distinct manufacturers. Since our work is focused on broadcast authentication, rather than key-sharing, we believe that integrating a handshake for the key exchange will have little theoretical merits while a proposal that could ease practical adoption is out of reach for the current work. For this reason, in what follows we simply assume that the corresponding keys exist on each node. For security reasons, we assume that new keys are negotiated at each protocol re-initialization, e.g., each time the car starts, and refreshed at periodic intervals, e.g., each hour/day depending on the intended security level. We do

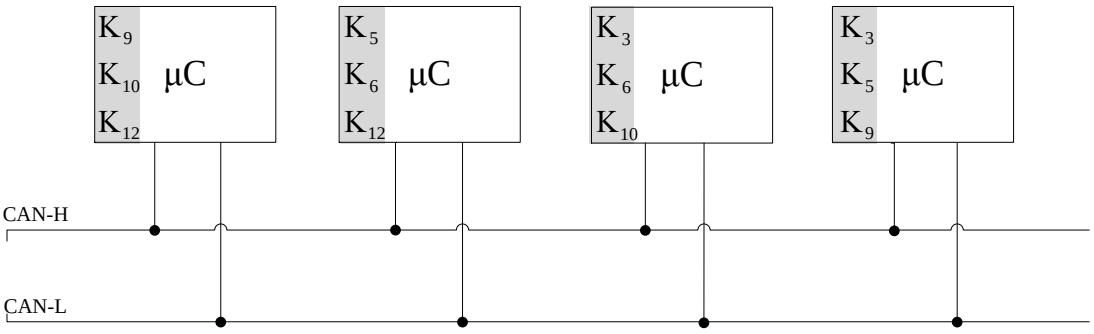


Figure 3.6: Setting for the main authentication scheme

however emphasize that while our implementation frequently uses truncated authentication tags to make them fit in the small CAN data-field (a procedure present in any of the related proposals), this does not make the authentication keys vulnerable since the keys can have full lengths, e.g., from 64–128 bits. It is just the probability that an adversary can simply guess an authentication tag that is higher due to the truncation. As previously discussed, re-initializing keys is also beneficial to assure freshness, thus the size of the message counter can dictate the frequency of reinitialization.

CONSTRUCTION 7. *(*LiBrA-CAN *- **Main Scheme)** Given an* M-MAC *construction for some security parameter ℓ and n nodes placed in groups of size g, we define protocol* LiBrA-CAN$_{\mathcal{N}^*}$(M-MAC, ℓ, n, g) *as the following set of actions for each CAN node denoted as $\mathcal{N}_i, i = 1..n$:*

1. Setup(ℓ, n, g) *is the key setup procedure. Let $t = \binom{n}{g}$ be the number of subsets of g nodes out of the n nodes, the* Setup *procedure generates the t random keys, each of ℓ bits, then distributes to each node the keys for the groups that he is part of. Let $\mathbb{K}^i_{\mathcal{N}} = \{\Bbbk^i_1, \Bbbk^i_2, ..., \Bbbk^i_{t'}\}$ with $t' = \binom{n-1}{g-1}$ denote the key set of each node $\mathcal{N}_i$ and $\mathbb{K}^{i,j}_{\mathcal{N}} = \{\Bbbk^i_1, \Bbbk^i_2, ..., \Bbbk^i_{t''}\}$ with $t'' = \binom{n-2}{g-2}$ the keys shared by each node $\mathcal{N}_i$ with node $\mathcal{N}_j$.*

2. SendMesTag$(\mathcal{N}_i, \widetilde{m}, \mathbb{K}^i_{\mathcal{N}})$ *on which node $\mathcal{N}_i$ whenever wants to broadcast a message $\widetilde{m}$ increments its local counter, computes the tag* M-MAC$_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m})$ *with its keyset $\mathbb{K}^i_{\mathcal{N}}$ and sends the message $\widetilde{m}$ and the authentication tag on the bus.*

3. RecMesTag$(\mathcal{N}_i, \mathcal{N}_j, \widetilde{m},$ M-MAC$_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m}))$ *on which node $\mathcal{N}_i$ receives a data frame containing message $\widetilde{m}$ from node $\mathcal{N}_j$ along with the corresponding tag* M-MAC$_{\mathbb{K}^j_{\mathcal{N}}}(\widetilde{m})$*. Node $\mathcal{N}_i$ checks if the message is fresh, i.e., counter up to date, and authentic for all common keys, i.e., $1 \leftarrow$ Ver$(\widetilde{m}, \Bbbk,$ M-MAC$_{\mathbb{K}^j_{\mathcal{N}}}(\widetilde{m}))$, $\forall \Bbbk \in \mathbb{K}^{i,j}$. If the tag is correct for all keys in the common keyset, i.e., $\mathbb{K}^{i,j}$, then message $\widetilde{m}$ is deemed authentic and the message counter updated.*

NUMBER OF GROUP, KEYS AND INFLUENCE OF CORRUPTED NODES. We try to give a quantitative justification over the influence of corrupted participants on the security of the scheme. For n participants we have 2^n groups, this includes one empty group and one group which contains all participants. There are $\binom{n}{g}$ possible groups of size g. For n nodes, given all groups of size g,

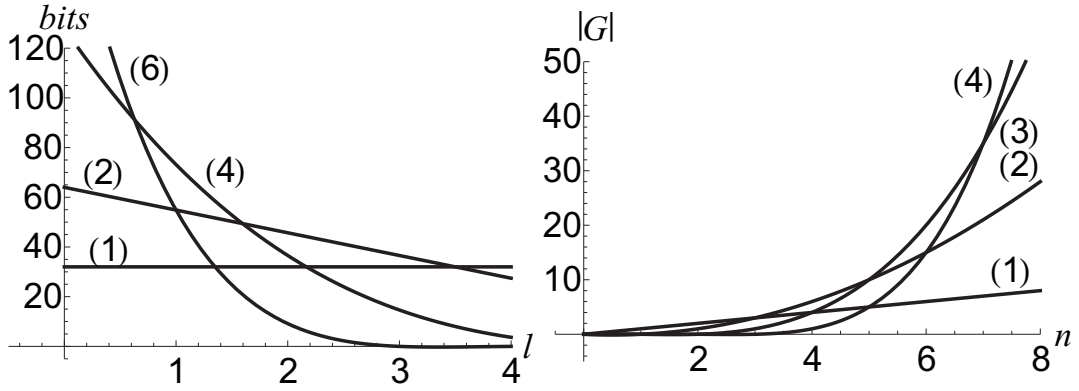


Figure 3.7: Fraction of recovered bits (left) for 8 nodes with $l = 1..7$ and number of groups (right) for $n = 1..8$ (group size given in the round brackets)

each node is part of $\binom{n-1}{g-1}$ groups. Further, if one considers that there is 1 corrupted node then he shares with each other node $\binom{n-2}{g-2}$ groups. If there are l corrupted nodes then they share with each node $\binom{n-l-1}{g-l-1}$ groups (this gives the number of corrupted keys on each node). For a more accurate view, we translate this discussion into security bits. Assume that the M-MAC is built by simple concatenation of regular MACs (each of them computed with the corresponding shared key). Having an M-MAC of some fixed bit-length t, each individual MAC is truncated to t/n bits. If keys are pair-wisely shared between the sender and receivers, i.e., $g = 1$, then each node receives exactly t/n security bits (since it is in possession of a single key that works for one of the MACs). But in case we share the keys between groups of size g, then each node is in possession of $\binom{n-1}{g-1}$ keys which is a fraction of $\binom{n-1}{g-1} \cdot \binom{n}{g}^{-1}$ from the total number of keys and equivalently from the bits carried by the M-MAC (obviously this more than $1/n$ for $g = 1$). Figure 3.7 shows on the left side the decrease in security bits, having fixed 256 bits for the tag, for groups of size $g = 1, 2, 4, 6$ in the case of $0..4$ adversaries (a dishonest minority). If we consider an authentication tag of 256 bits, if this is to be shared by 8 nodes with 8 different keys then only $256/8 = 32$ bits per node remain. Contrary, in the case of groups of size $g = 2$ there are 64 bits for each node; even if there is 1 corrupted node, still 54 bits remain untouched (while this is not much, it may be enough for real-time security). Generally, with 1 adversarial node the highest security is for $g = n/2$ due to the binomial expansion in which the middle coefficient is the larger. With at most $n/2$ adversaries, i.e., adversarial minority, the highest security is at $g = 2$ and it decreases linearly. On the right side, Figure 3.7 shows the number of groups (which translates into keys) in the case of $n = 1..8$ nodes and subgroups of size $g = 1, 2, 3, 4$. Obviously, this grows exponentially, but for smaller g the number of keys is decent and gives strong security benefits.

3.3.3 Variations of the main scheme

Variations of the main scheme are now presented based on the extended version of our conference report [30]. We further designed several variations of the main authentication scheme that give efficient trade-offs as shown in the experimental results section. These variations can be roughly grouped in two main classes. First, for non-homogeneous networks it makes sense to have a *master oriented authentication* where the node with higher computational power deals with the main part of the authentication procedure. Second, for homogeneous networks where all nodes have similar computational power it makes sense to have a *distributed authentication* scheme where all

participants contribute to the authentication task. We enumerate all these constructions in what follows.

1. **CENTRALIZED AUTHENTICATION**. A master oriented communication makes sense since it is practical to have one node with higher computational power that can take care of the most intensive part of the authentication. This is also supported by our experimental results. More, if the master node is a trustworthy third party, there are clear security benefits if he handles all keys that are shared between nodes since he can continue the authentication further with all the remaining keys (not only with the keys known to the sender). This is summarized by the next construction. Figure 3.8 shows the master node and the slave nodes connected to the bus, it also outlines the keys that are shared between nodes.

   CONSTRUCTION 8. *(Centralized Authentication) Given an* M-MAC *construction for some security parameter ℓ and n nodes placed in groups of size g, protocol* $\mathsf{CN\text{-}LiBrA\text{-}CAN}_{\mathcal{M},\mathcal{N}^*}(\mathsf{M\text{-}MAC}, \ell, n, g)i$ *defined as follows. Let $\mathbb{K}_{\mathcal{M}}$ denote the keyset of the master node and run the previous key-setup procedure only for the master node (i.e., the master places the slaves in groups of size g and shares keys with the groups). The following set of actions hold for the master $\mathcal{M}$:*

   1. $\mathsf{RecMesTag}(\mathcal{M}, \mathcal{N}_i, \widetilde{m}, \mathsf{M\text{-}MAC}_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m}))$ *on which master $\mathcal{M}$ receives message $\widetilde{m}$ and authentication the tag* $\mathsf{M\text{-}MAC}_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m})$ *from slave $\mathcal{N}_i$. Subsequently, master $\mathcal{M}$ checks if the counter is up-to-date and if the message is authentic, i.e.,* $1 \leftarrow \mathsf{Ver}(\widetilde{m}, \Bbbk, \mathsf{M\text{-}MAC}_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m}))$, $\forall \Bbbk \in \mathbb{K}^i_{\mathcal{N}}$. *If so, he proceeds to authenticating the tag to other nodes with* $\mathsf{SendTag}(\mathcal{M}, \widetilde{m}, \mathbb{K}^i_{\mathcal{N}})$ *and updates the message counter.*

   2. $\mathsf{SendTag}(\mathcal{M}, \widetilde{m}, \mathbb{K}^i_{\mathcal{N}})$ *on which master $\mathcal{M}$ gathers all the remaining keys $\mathbb{K}_{\mathcal{M}} \setminus \mathbb{K}^i_{\mathcal{N}}$ computes* $\mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{M}} \setminus \mathbb{K}^i_{\mathcal{N}}}(\widetilde{m})$ *and broadcasts it as an authentication frame with the same ID as the original message (note that in this case the* M-MAC *is computed with the remaining $\binom{n}{g} - g$ keys, there is no restriction from the construction of* M-MAC *to do it so).*

   and for each of the slaves $\mathcal{N}^*$:

   1. $\mathsf{RecMesTag}(\mathcal{N}_i, \mathcal{N}_j, \widetilde{m}, \mathsf{M\text{-}MAC}_{\mathbb{K}^j_{\mathcal{N}}}(\widetilde{m}))$ *on which slave $\mathcal{N}_i$ receives message $\widetilde{m}$ and an authentication tag from another slave $\mathcal{N}_j$ and proceeds similarly to master $\mathcal{M}$ by checking if the message is authentic but only with respect to the keys $\Bbbk \in \mathbb{K}^i_{\mathcal{N}} \cap \mathbb{K}^j_{\mathcal{N}}$ that he shares with slave $\mathcal{N}_j$. The message is not deemed authentic until a successful* $\mathsf{RecTag}(\mathcal{N}_i, \mathcal{M}, \mathcal{N}_j, \mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{M}} \setminus \mathbb{K}^j_{\mathcal{N}}}(\widetilde{m}))$ *event follows.*

   2. $\mathsf{RecTag}(\mathcal{N}_i, \mathcal{M}, \mathcal{N}_j, \mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{M}} \setminus \mathbb{K}^j_{\mathcal{N}}}(\widetilde{m}))$ *on which slave $\mathcal{N}_i$ receives an authentication frame containing the tag* $\mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{M}} \setminus \mathbb{K}^j_{\mathcal{N}}}(\widetilde{m})$ *from the master $\mathcal{M}$ (that continues the authentication of slave $\mathcal{N}_j$) and verifies for all keys $\Bbbk \in \mathbb{K}^i_{\mathcal{N}} \cap \left(\mathbb{K}_{\mathcal{M}} \setminus \mathbb{K}^j_{\mathcal{N}}\right)$ if the tag is correct. If for all keys in its keyset the tag is correct then message $\widetilde{m}$ is deemed authentic and the message counter updated.*

   3. $\mathsf{SendMes}(\mathcal{N}_i, m, \mathbb{K}^i_{\mathcal{N}})$ *on which slave $\mathcal{N}_i$ whenever wants to broadcast a message $\widetilde{m}$ increments its local counter, computes the tag* $\mathsf{M\text{-}MAC}_{\mathbb{K}^i_{\mathcal{N}}}(\widetilde{m})$ *with its keyset $\mathbb{K}^i_{\mathcal{N}}$ and sends the data frame containing message $\widetilde{m}$ and the corresponding tag on the bus.*

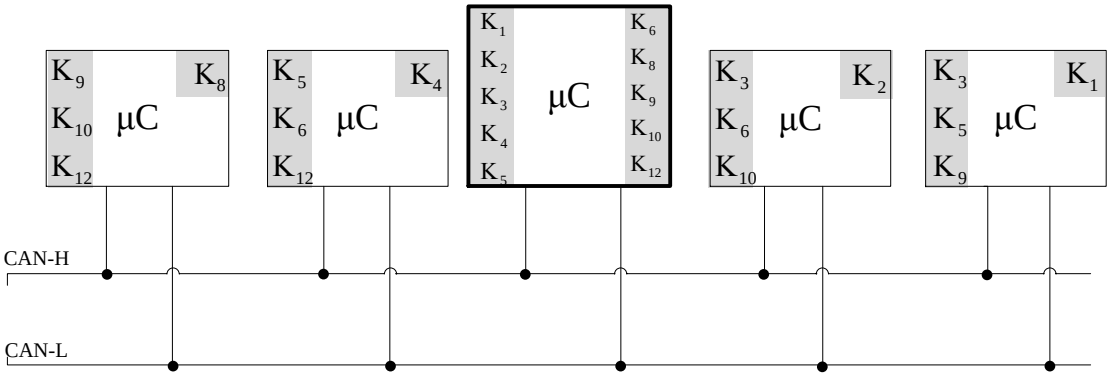


Figure 3.8: Master and slave microcontrollers (μC) in a setting for centralized authentication

2. **CUMULATIVE AUTHENTICATION**. Since in some scenarios small delays may be acceptable, we can take benefit of them and increase the efficiency of the main scheme. In the *cumulative authentication* scheme a timer can be used and all messages are accumulated by the master over a predefined period δ then authenticated at once (this procedure can be employed in the slave-only settings as well). While this introduces an additional delay δ, similar to the case of the TESLA protocol, this delay can be chosen as small as needed to cover application requirements. Distinct to the case of the TESLA protocol the delay is not strongly constrained by external parameters (such as oscillator precision, synchronization error, bus speed, etc.).

3. **LOAD BALANCED AUTHENTICATION**. The *centralized authentication* scheme is beneficial in the case when the communication master has higher computational resources, but it may be the case that the master node is already busy with other computational tasks. For such case, a *load balanced* version of the scheme can be used in which the communication master can send a flag (authenticated along the message) to point for a particular slave(s) to carry the authentication further.

4. **CASCADE AUTHENTICATION**. Indeed, an authentication master may not always be present. Moreover, several events can lead to his unavailability (for example he can enter Bus Off-mode due to problems with the transceiver or the ECU itself can suffer a malfunction). For this purpose we introduce the cascade authentication scheme where the slaves reply in a cascade manner by sending the authentication tag for their group of keys. In what follows, we assume that all the nodes continue the authentication in a round-robin fashion until they reach the sender (or stop after sufficient authentication tags are released). Thus, we point out to node $(i+1)$ as the next node and whenever we reached the n-th node the first one becomes the next, etc.

   CONSTRUCTION 9. *(Cascade Authentication) Given an* M-MAC *construction for some security parameter ℓ and n nodes placed in groups of size g, we define protocol* DC-LiBrA-CAN$_{\mathcal{N}^*}$(M-MAC, ℓ, n

as follows. Let $\mathbb{K}_{\mathcal{N}}^{i,j}$ denote the keys shared between nodes i and j (we assume the same key-setup as previously, except that the master does not play any role in authentication), then the following set of actions is defined for each of the nodes $\mathcal{N}^*$:

1. $\mathsf{RecMes}(\mathcal{N}_i, \mathcal{N}_j, \widetilde{m})$ *on which node $\mathcal{N}_i$ receives a data frame containing message $\widetilde{m}$ from another node $\mathcal{N}_j$ checks if the counter is up-to-date then stores the message in a queue of messages to be authenticated.*

2. $\mathsf{RecTag}(\mathcal{N}_i, \mathcal{N}_j, \mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{N}}^{j,j+1}}(\widetilde{m}))$ *on which $\mathcal{N}_i$ receives an authentication frame containing tag* $\mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{N}}^{j,j+1}}(\widetilde{m})$ *from another node $\mathcal{N}_j$ and verifies for all keys $\Bbbk \in \mathbb{K}_{\mathcal{N}}^{i} \cap \mathbb{K}_{\mathcal{N}}^{j,j+1}$ if the tag is correct. If for all keys in its keyset a correct tag was received then message $\widetilde{m}$ is deemed authentic and the message counter updated. If $i = j + 1$ then it proceeds to* $\mathsf{SendTag}(\mathcal{N}_i, \widetilde{m}, \mathbb{K}^i)$.

3. $\mathsf{SendTag}(\mathcal{N}_i, \widetilde{m}, \mathbb{K}^i)$ *on which node $\mathcal{N}_i$ gathers all the keys shared with node $\mathcal{N}_{i+1}$ in the set $\mathbb{K}_{\mathcal{N}}^{i,i+1}$, computes* $\mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{N}}^{i,i+1}}(\widetilde{m})$ *and broadcasts it as an authentication frame.*

4. $\mathsf{SendMes}(\mathcal{N}_i, \widetilde{m}, \mathbb{K}_i)$ *on which node $\mathcal{N}_i$ whenever wants to broadcast a message $\widetilde{m}$ increments its local counter, computes the tag* $\mathsf{M\text{-}MAC}_{\mathbb{K}_{\mathcal{N}}^{i,i+1}}(\widetilde{m})$ *and sends the data frame containing $\widetilde{m}$ followed by an authentication frame containing the tag on the bus.*

5. **TWO-STAGE AUTHENTICATION**. In the case of *two-stage authentication* we assume a scenario with nodes of equal computational power. In this case each node can start broadcasting by sending a tag which includes only a part of the keys for the subgroups that he is part of and a second node (pointed out by some flag, or predefined in protocol actions) continues with the authentication. The procedure is repeated until the desired number of authentication frames is reached.

6. **MULTI-MASTER AUTHENTICATION**. For the same reasons, a distributed version of the *centralized authentication* scheme can be imagined. In this case, several nodes with higher computational power can form a group of communication masters. Each of them may broadcast a distinct authentication tag and if any such tag is missing, due to the unavailability of a particular node, the other masters will take care of replacing this tag with one of their own.

3.4 Proposed protocols, a comparison [4]

This section contains a performance discussion that covers (in part) all of the recently proposed protocols (including our contributions from the previous sections) for assuring authentication on in-vehicle networks .

[4]The results presented in this section are based on author's previously published work: Paula Vasile, Bogdan Groza, and Stefan Murvay. *Performance analysis of broadcast authentication protocols on CAN-FD and FlexRay*, WESS: 10th Workshop on Embedded Systems Security (affiliated to ESWEEK 2015), ACM, 2015.

3.4.1 Revisiting protocols from a keying perspective

We extract here our main results from [95]. While in the previous subsection we encountered several proposals, the crux of the problem (and the differences between the previous schemes) comes from the way in which the keys are distributed between the nodes. It is clear that only *symmetric key* primitives can be used, due to restrictions on computational power and bandwidth, and thus the number of authentication tags (which determines the bus-load) comes from the way in which keys are allocated. We now revisit the previous proposals and classify them based on the keying procedure they employ. Mutatis mutandis, all of the previous schemes fit in one of the following four paradigms:

1. *Single authentication key* is the simplest keying procedure in which all frames are authenticated with a single key. In this case, all senders and receivers must be in possession of the authentication key and if one of them is corrupted all security is lost. Since each frame carries a single tag, this protocol provides a baseline for the bus-load (from this perspective both CANAuth [91] and CaCAN [47] fit into this paradigm). *ID-oriented keying* is a variation of this in which each frame carries a single authentication tag, but the key to compute this tag is unique for each of the IDs. This procedure is explored in CANAuth [91]. As already noted in previous work, usually there are too many IDs to have a unique key for each of them, but this requirement can be relaxed by having a unique key for each group of frames that is selected based on a predefined mask. From the bus-load perspective this protocol has identical requirements to the previous (a single tag again) and thus it matches the baseline for the bus-load when a single authentication key is used.

2. *Pairwise keying* is the rather natural way in which unique authentication keys are shared by each distinct pair of nodes. This procedure is also employed in MaCAN [35]. However, in case of n nodes this leads to $\frac{n(n-1)}{2}$ keys and $n-1$ authentication tags. For higher number of nodes, the overhead is unlikely to be handled by the available bandwidth and computational power available in automotive networks. For this reason, in related work, e.g., [35], it was already suggested that nodes that share the same trust level can use the same secret key for authentication. In some sense, this opens door for the next procedure.

3. *Group keying* is an improvement over pairwise key sharing that allocates keys over groups of nodes. The main ideea is to build groups that have an intermediate size between 2 (which corresponds to pairwise keying) and n (which corresponds to having a single authentication key in case of n nodes). The security level is higher if malicious nodes are in minority, the procedure is explored in LiBrA-CAN [28]. Worst than in the previous case, the number of keys and tags grows exponentially, but again if more nodes are grouped under the same key (as in the case of pair-wise keying) then the number of keys stays low.

4. *TESLA-like keying* requires authentication keys to be broadcast in a periodic manner, i.e., at fixed time intervals. This means that besides the regular frames that are sent over the network a frame containing the key is released at fixed intervals. However, the problem of the protocol is not necessarily in the additional overhead but in buffering the received frames until the authentication key is received and in the intrinsic authentication delay.

For the case of shared keys, i.e., all protocols except TESLA, in order to draw a synthetic comparison we need to fix the following parameters for a setup with n receivers in groups of size g:

- the total number of keys:

$$\mathcal{K} = \binom{n}{g},$$

- the total number of keys stored on a single node which is also the number of tags computed by the node when sending a message to all of the other nodes:

$$\mathcal{K}_{\mathsf{SendMes}} = \binom{n-1}{g-1},$$

- the total number of tags intended for a single receiver which is also the computational load on the receiver side:

$$\mathcal{K}_{\mathrm{recv}} = \binom{n-2}{g-2},$$

- the fraction of tags intended for a single node out of the total number of tags:

$$\mathrm{F}_{\mathrm{recv}} = \binom{n-2}{g-2}\binom{n-1}{g-1}^{-1} = \frac{\mathcal{K}_{\mathrm{recv}}^{node}}{\mathcal{K}_{\mathsf{SendMes}}^{node}},$$

- the size of the tag for a security level of ℓ bits:

$$\mathrm{S} = \ell\binom{n-2}{g-2}^{-1}\binom{n-1}{k-1} = \ell \cdot \mathrm{F}_{\mathrm{recv}}^{-1},$$

- the fraction of uncorrupted tags for a single receiver in case of m corrupted nodes:

$$\mathrm{F}_{\mathrm{recv}}^{\mathrm{corr}} = \frac{\binom{n-2-m}{g-2}}{\binom{n-1}{g-1}},$$

- the security level in case of m corrupted nodes which is the fraction of uncorrupted security bits for a single receiver:

$$\ell^{\mathrm{corr}} = \mathrm{S} \cdot \mathrm{F}_{\mathrm{recv}}^{\mathrm{corr}}.$$

In this formalism, the case when the size of the subgroup is $g = n$ corresponds to the case of a single authentication key while $g = 2$ corresponds to the case of pair-wise key sharing. In terms of bandwidth, the ID-oriented keying and TESLA-like keying overlaps with the case of a single authentication key. We can now draw a synthetic comparison.

In Figures 3.9 and 3.10 we depict the size of the tag as well as the remaining uncompromised bits in case of 1 corrupted node with $n = 8$. It is easy to note that when a single key is used, i.e., $n = 8, g = 8$, the size of the resulting authentication tag is kept at a minimum, i.e., 128 bits for a security level of 128 bits. In the same case however, if a single node is corrupted the number of uncorrupted bits drops to 0; since all nodes share the same key. Sharing the keys pair-wisely leads to no security loss in case of 1 corrupted node, as each two nodes share a distinct key, however for a security level of 128 bits the tag quickly grows to 896 bits (the plot is truncated at 250 bits since

tags larger than this are clearly not suited for real-time communication on an embedded network). Finally, having groups of size 3 or 4 gives a nice trade-off between these values.

Figure 3.11 compares the number of keys on each node, tags computed by each node and tags verified by each node. For the extreme case of group size 2 and 8 these values are the lowest but with the aforementioned disadvantage that either the tag is too large, i.e., $g = 2$, or security is lost when one node is compromised, i.e., $g = 8$.

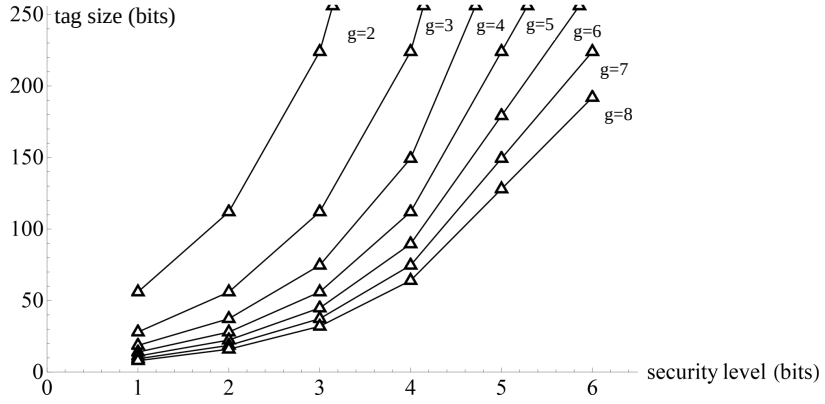


Figure 3.9: Tag size for a security level of 16, 32, 64 and 128 bits with n=8 and g=2,3...8 (as depicted in [95])

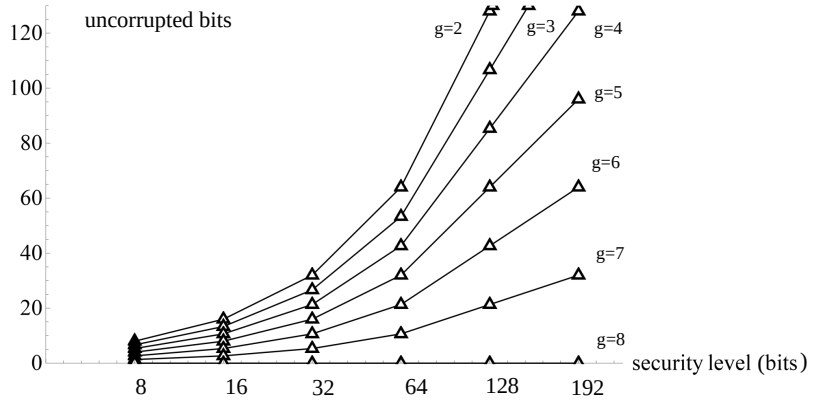


Figure 3.10: Uncorrupted bits in case of 1 corrupted node for a security level of 16, 32, 64 and 128 bits with n=8 and g=2,3...8 (as depicted in [95])

3.4.2 Overview of experimental results

The detailed experimental results are available in [29], [31], [30] and [95]. Here we will only make a brief overview over the main results on computational load and bus-load, either depicted experimentally or by CANoe simulations.

COMPUTATIONAL LOADS. The computational resources on automotive-grade microcontrollers are sufficient to handle cryptographic primitives. These may indeed vary from low-end to high end ones as can be seen in Table 3.2. There are no doubts that for real-world deployments the

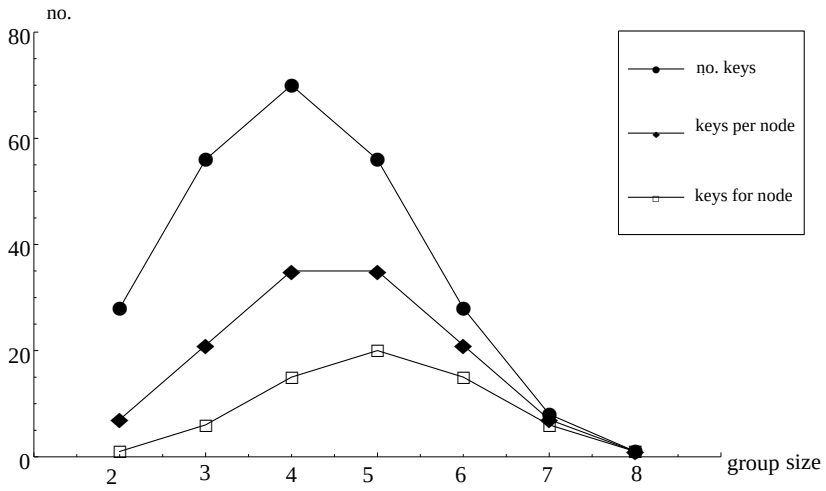


Figure 3.11: Comparison on the number of keys, number of computed tags and number of verified tags with n=8 and g=2,3...8 (as depicted in [95])

Table 3.2: Technical specifications for S12XDT512 and TriCore TC1797

Characteristic	**S12XDT512**	**TriCore TC1797**
Flash Memory	512 KB	4 MB
RAM	20 KB	156 KB (SRAM)
EEPROM	4 KB	16 KB emulated by 64KByte data Flash
Clock Speed	80 MHz	180 MHz

protocols should rely on hardware implementations. So far hardware support is missing from the devices that we work with, but this will clearly change in the short run. In Table 3.3 we depict some computational results for the current cryptographic hash standard SHA256 as well as for the less intensive SHA1 and MD5 (which are insecure for current demands, but they are enough for real-time security). All these show that there is enough computation support for the primitives that our protocol requires.

BUS-LOAD, EXPERIMENTAL. For the bus-load of TESLA-like schemes we refer the reader to [29], we do not present the results here as we do not consider the scheme viable due to the delays that it introduces and buffering requirements. Similarly, we discard the one-time signature based

Table 3.3: Execution time for some common hash functions (as determined in [30])

Function	**Input size (bytes)**					
	S12X			TriCore		
	0	16	64	0	16	64
MD5	$371\mu s$	$374\mu s$	$689\mu s$	$10.16\mu s$	$11.00\mu s$	$18.34\mu s$
SHA1	$1.144ms$	$1.148ms$	$2.285ms$	$14.64\mu s$	$15.10\mu s$	$27.60\mu s$
SHA256	$2.755ms$	$2.755ms$	$5.440ms$	$41.70\mu s$	$42.35\mu s$	$80.80\mu s$

Table 3.4: Results for centralized authentication with $n = 4, g = 2$ (as determined in [30])

Master	Slave	δ	Bitrate	Bus load
S12X	4xS12X	$2.54ms$	125 kbps	53.84%
PC	4xS12X	$1.848ms$	125 kbps	72.22%
TriCore	4xTriCore	$267\mu s$	1 Mbps	54.31%
PC	4xTriCore	$378\mu s$	1 Mbps	42.54%

schemes as they induce too high computational loads. Here we only present the experimental results from [30] regarding the bus-load of LiBrA-CAN in Table 3.4. The results are only for the case of centralized authentication since existing development boards do not support larger CAN-FD frames and we cannot fit the authentication payload on the 64 bit data-field of CAN. Thus we rely on a master node for authentication. We present results for CAN-FD by CANoe simulation in what follows.

BUS-LOAD, BY CANOE SIMULATION. The CANoe based simulation has a convincing word on the feasiblity of these protocols for the current bandwidth available for in-vehicle networks. In [95] we considered a network with 30 ECUs and we grouped them to obtain uniform bandwidth for each group as suggested in Figure 3.12. The results in Table 3.5 show that group keying, i.e., LiBrA-CAN, is the top performer. Indeed all other schemes are nicely accommodated by the bandwidth available on modern CAN-FD and FlexRay network layers.

Authentication protocol	Recorded bus-load: CAN-FD [%]	Recorded bus-load: FlexRay [%]
Baseline	43.36	58.55
Single authentication key	48.97	58.57
TESLA-like keying 10 ms	64.97	71.61
TESLA-like keying 20 ms	57.40	65.89
TESLA-like keying 40 ms	53.62	61.56
TESLA-like keying 80 ms	51.73	59.41
Group keying (groups of 2)	68	58.57
Group keying (groups of 3)	82.21	58.57
Group keying (groups of 4)	70.51	58.57

Table 3.5: Recorded busload on CAN-FD and FlexRay as reported in [95]

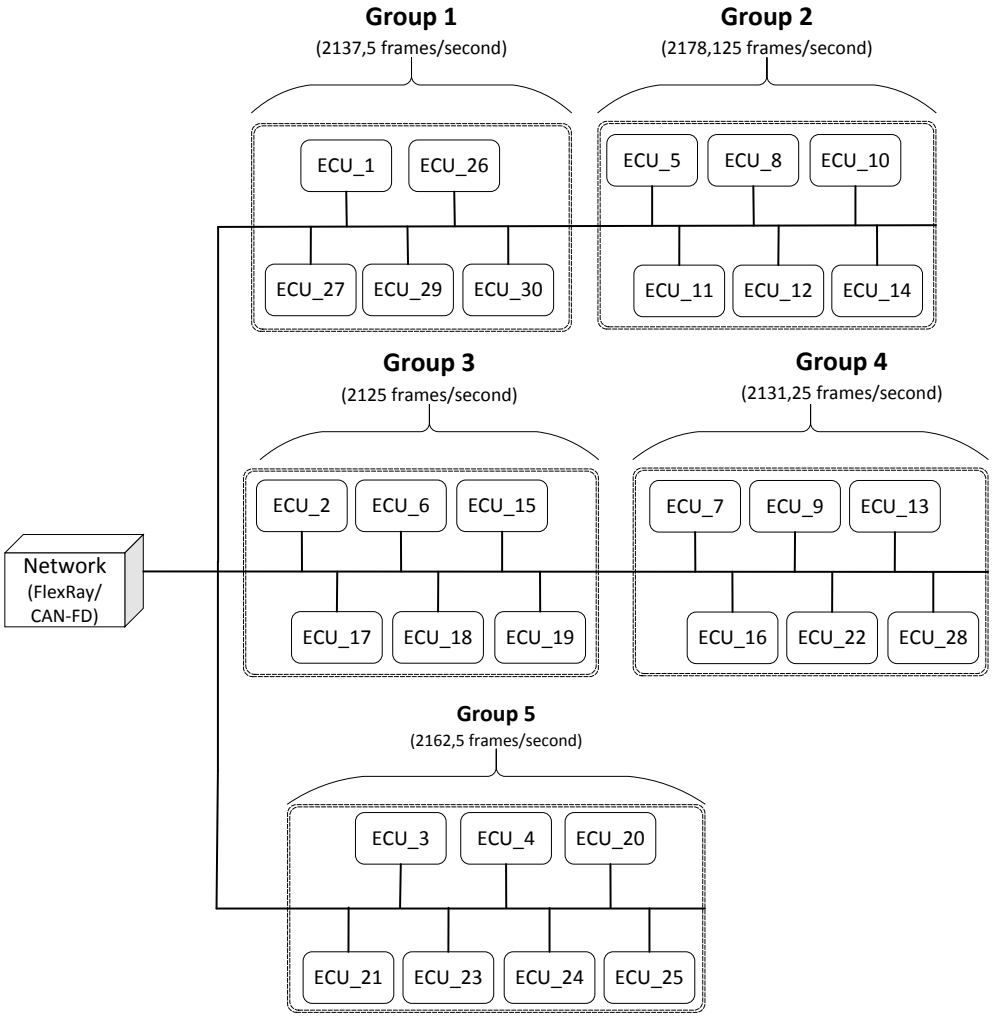


Figure 3.12: The groups of ECUs as used in our experiments from [95]

Chapter 4

Cryptography on wireless interfaces

This section documents our research results on the security of wireless interfaces inside vehicles. While the security of wired in-vehicle networks, e.g., the CAN bus, was within our focus since 2008, wireless interfaces came to our attention much more recently, since 2012. As expected, given the much shorter time frame and possibly a more demanding scenario (wireless interfaces are dispersed and more exposed to adversaries) we are here only at our first research steps. In what follows we address TPMS systems (Tire Pressure Monitoring System) and the vehicle access through wireless keys that are paired with mobile phones over NFC. The first topic was subject to a recent conference report while the second was presented at an industry conference only as an initial overview for the security design (there is much more work that will follow). While not a direct part of the body of wireless security issues, we also include here one of our collaborations with the industry on generating randomness on automotive grade controllers. This subject seems to be particularly problematic for TPMS sensors and wireless keys as our research, as well as other reports, proved that randomness is problematic in many automotive applications (e.g., in challenge-response protocols for RF keys).

4.1 Protocols for wireless interfaces: TPMS sensors [1]

We first review the standard TPM system functionalities and reported attacks as presented by us in [76]. The standard functionality for tire sensors is to monitor the pressure inside the tire which further impacts on at least three directions: i) reducing tire wear and fuel consumption, ii) reducing CO_2 emissions (comforting the environment) and iii) improves the breaking distance and the control of the vehicle, thus having a positive effect on the safety of drivers and passengers. While a recent study shows the benefits of TPMS in reducing CO2 emissions to be only marginal [94], for safety and comfort the advantages of the system are undisputed. The TPMS system became mandatory in the US since 2008 and in the EU since 2014.

In practice, there are two distinct kinds of implementations: i) direct systems (the common choice and the subject of our work) in which a sensor is located inside the tire and ii) indirect systems which rely on information from the ABS (improperly inflated tires lead to detectable differences when breaking, but such systems are inaccurate and not commonly used). While some TPMS deployments were initially based on four receivers, one nearby each wheel, the current

[1] The results presented in this section are based on author's previously published work: Cristina Solomon, Bogdan Groza, *LiMon - lightweight authentication for tire pressure monitoring sensors*, 1st Workshop on the Security of Cyber-Physical Systems (affiliated to ESORICS'15), 2015.

trend is to have a single receiver that gathers the frames from each wheel sensor. Figure 4.1 depicts a system with a single receiver and four wheel sensors which is of concern to our research here.

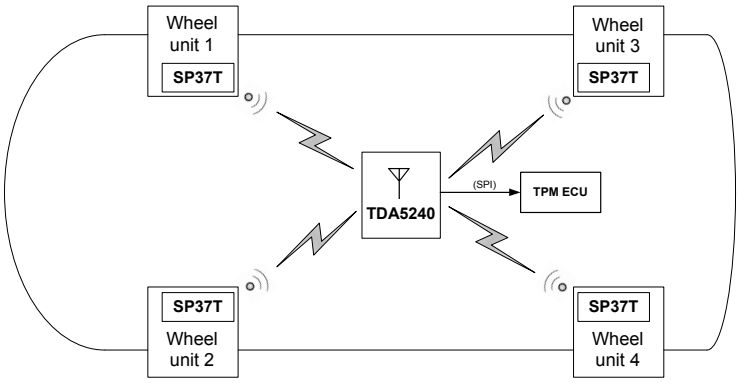


Figure 4.1: TPM system with Infineon SP37 sensors and TDA5240 receivers (as used in our work [76])

SECURITY ISSUES. Despite the fact that the TPMS is part of the safety system of the car since, security mechanisms are completely absent from such systems. Recent research works [42] were quick in determining several attacks on such platforms:

- eavesdropping can be easily performed at a range of 10m (and with improved instrumentation at up to 40 m),
- vehicles can be tracked by using the ID that is broadcast by the sensors,
- packets can be injected in the system, triggering false alarms on the display (beside the obvious discomfort, these may determine the driver to stop the car for inspection which could cause further incidents),
- battery drainage is possible if there are packets that could trigger an immediate response from the sensors.

The attacks presented in [42] require just an average adversary that is in possession of some easy to find radio equipments. The recent patent application for TPMS security from Continental [90] proves that the industry is aware and concerned by such attacks. The patent suggests that the main concern is on the fact that the driver can be determined to stop the car in case that an adversary injects data claiming that the tires are under-inflated [90] (a similar scenario is also outlined in the work that initially showed these attacks [42]). Clearly, threats can become much more serious in the near future if information from the TPMS sensors will be also actively used by breaking or stability controls inside the car. As a consequence to these, opening road for research in this direction is clearly necessary.

4.1.1 Analysis of previous proposals

Despite the myriad of papers addressing security in wireless sensors network, to the best of our knowledge there are only two previous proposals for assuring security in TPMS sensor systems. One is from the academic research community [102] and the second is the patent application from Continental [90]. We do present here our analysis of these proposals as done by us in [76]:

1. *Continental's patent* EP2719551 [90] appears to be the first attempt from the industry to address TPMS security issues. At the very least the patent shows that the industry is aware of the problem and the solution can be viewed as an improvement to the no-security option. On the negative side, the solution completely fails in assuring security as the proposed mechanism can be trivially broken, moreover, the brief security analysis provided in the patent application is wrong (as we discuss next). The solution works as follows: two PRNGs are used (both implemented around a CRC polynomial) one for the wheel unit, the other for the receiver. Since there are four wheels, each wheel has its own PRNG built on the same CRC code but initialized with a distinct seed, while the receiver keeps 4 instances, one for each of them. A seed, generated from data that is exchanged between the wheel units and the receiver at vehicle start-up, is used as input to the PRNG and the output (referred as authentication marker) is XORed with the CRC of each message in order to assure authentication. We now enumerate the problems in this proposal starting with the more severe ones:

   (a) CRCs (Cyclic Redundancy Checks) are used to build an *authentication marker* but it is commonly known that while CRCs can assure integrity checks (against unintentional errors) they fail in assuring authenticity (even if one embeds a secret key in the construction) because they are linear transformations.

   (b) The secret seed can be easily found. The patent is not really clear on the size of the underlying CRC code, however under paragraph [0034] it claims that it can be very light, 8 bits being sufficient since for an emission period of 60 seconds since to observe the full period of 256 rounds one would need 4.25 hours. This claim actually misses the fact that the CRC, being a LFSR, can be broken in much shorter time and one does not need to record all the possible outputs.

   (c) The key-sharing procedure for exchanging the seed between the wheel units and the central receiver has no protection at all. Since there is no cryptography implemented on the nodes, the seed value is exchanged unencrypted when the system starts. This makes it easy to mount an attack if the adversary witnesses the communication right from the beginning.

   (d) A final problem are the ambiguities in the patent. For example the patent claims that "at least a part of each data packet is encoded with an authentication marker" but fails to make it clear which part. The same ambiguities are in describing the exact security level that is expected, i.e., the size of the secret seed, the CRC polynomial, etc.

2. *Xu et al.* [102] present a protocol that is well designed, particularly targeting spoofing and tracking attacks. The core of the protocol is built around the 32-bit symmetric encryption algorithm KATAN and uses CBC-MAC to assure message authentication. Additionally, a LFSR is used to generated sequential numbers that look pseudo-random in order to prevent replay-attacks (the use of a simple counter is avoided in order to prevent tracking of the device, as sequence numbers are predictable). To ensure privacy, as vehicles can be tracked by the use of the sensor's ID, a pseudo-ID is randomly generated before the generation of each session key. A proof-of-concept implementation is provided on an Arduino based platform. The shortcoming that we see for this proposal is that it is deployed on a platform that does not matches the real-world deployments, our experimental analysis showed that KATAN is too intensive for real-world TPMS sensors. We give more details on the real-world system specification in the following section.

As one could easily note, the proposals are situated on two extremes. The patent [90] tries to build an inexpensive solution by garnering what already exists on the sensors but fails to result in a secure protocol. Xu et al. [102] provide a rigorous security design but the experimental results are obtained on an Arduino platform which makes it unclear if the solution can be readily deployed on real world TPM systems. Our work tries to bridge between the two worlds by using real world components and a rigorously designed protocol.

4.1.2 Protocol design (LiMon)

A brief encounter of our design goals now follows. *Lightweight cryptography* was a first design step, we make use of some of the lightest designs published so far SPECK and PRESENT [6]. Hardware implementations of cryptographic primitives are missing from the sensors. In the results section of our conference report [76] we provide quantitative measurements on the performance of software implementations for these primitives. We keep *confidentiality and privacy just as an option* to ensure a lightweight protocol. We opt for these since privacy does not appear to be the main concern (there are so many ways to track a car, e.g., the license plate) while energy consumption and authenticity are more critical. We keep frame encryption, which implies hiding the ID as a second option in our protocol. An *ISO based key-exchange* (for establishing the session keys) rather than re-inventing a protocol per paper seems to be a the preferred choice especially when we address an area standardized by the industry. *Energy efficiency* is not the last goal but in fact the source of two of the previous objectives which triggered the use of light-weight cryptography and keeping extra functionalities just as an option.

The protocol that we design in [76] accounts for several procedures that we outline in what follows:

1. *Master key establishment* is the procedure for fixing a shared key between the sensor and receiver. This proves to be a difficult issue because we cannot rely on public-key infrastructure (too many computational constraints at the sensor level). Xu et al. [102] propose for the initialization of the secret master key to be triggered only by the sensor and only during rare events, e.g., when the tire is inflated for the first time. This has drawbacks in case when the driver spontaneously needs to chance the wheel, e.g., in case of a flat tire. In Continental's patent [90], the key is exchanged in clear-text each time the vehicles starts. This procedure is easy to implement but also easier to eavesdrop, nullifying the security of the protocol that follows. To bring some clarity over these alternatives, our work tries to make a separation between two larger methodologies. The first gathers both the proposals from [102] and [90] while the second is an alternative that we propose in [76]:

    (a) *The resurrecting duckling* is an idea capsed in the seminal work of Stajano [77]. The idea of imprinting comes from a metaphor inspired by biology: a duckling emerging from its egg recognizes as its mother the first moving object it sees that makes a sound, a phenomenon called imprinting. The procedures in the Continental's patent [90] and the ones from the work of Xu et al. [102] can be viewed as particular cases of the *imprinting* phenomenon. In all situations we assume that imprinting pairs the sensor with the first device that sends it a key. We further separated between three distinct cases of imprinting. *Rare event imprinting* which is the case when the key is imprinted when a rare event takes place. *External tool imprinting* when the key is imprinted by the use of an external tool via a secure communication medium, e.g., OBD (On-Board

Diagnosis) or Bluetooth. *Safe environment imprinting* which requires the key to be imprinted in a secure environment, e.g., at an authorized garage, a case in which the key can be simply sent in plaintext.

(b) *Environmental data.* We should not forget a relevant aspect, all these sensors are located in the same environment which can provide a rich common entropy that is hard to be guessed by outsiders. First, accelerometer data is available both to the internal ABS unit as well to the wheel sensor. This data was previously used to learn the wheel on which a tire is connected, but can be as well used to generate a shared key. Generally, vibrations from the environment can be easily captured and they should be similar for devices located on the same vehicle. There is extensive literature on using accelerometer data to generate a shared key. Since this key will not provide an exact match on the sender and receiver side, fuzzy cryptography [20] can be applied to correctly extract a key.

2. *Session key establishment* is a procedure for exchanging a fresh session key each time the protocol starts or whenever the session counter reaches the maximum value. Given the industry driven nature of the application area, it is best to stay to current standards in order to make such solutions viable for adoption by manufacturers. The ISO/IEC 9798 provides a well known family of authentication protocols based on both symmetric and asymmetric primitives. Recently, this standard was subject to rigorous formal analysis which lead to several improvements and fixes [4]. Once the master-key is shared between the sensor and receiver, the session key-establishment can be done with any mutual authentication protocol. For example, the ISO 9798-2-4 three-pass mutual authentication scheme. In Figure 4.2 we depict the structure of this key-exchange protocol. The notations are the expected ones: $\Bbbk_{ms}$ denotes the master key shared between the sensor and the receiver, N is the regular notation for cryptographic nonces, i.e., random values, while $\mathsf{Sens}_{\mathsf{ID}}$ and $\mathcal{R}$ are the identities of the sending sensor and of the central receiver. The session key $\Bbbk_{ses}$ can be simply derived from the exchanged values $N_{\mathsf{Sens}_{\mathsf{ID}}}, N_{\mathcal{R}}$ and the master key $\Bbbk_{ms}$ with the help of some key derivation process. This incurs only a small computational cost in the order of an additional encryption and can be straight-forwardly built upon the primitives that we deploy for the rest of the protocol.

3. *Frame authentication* is done with the standard CBC-MAC based on SPECK as outlined in Figure 4.3 (i). This construction is secure as the data field has a fixed size of 64 bits in our example (the length of the full frame is 96 bits since 32 bits are added for the authentication tag). The session key $\Bbbk_{ses}$is used via some standard key derivation technique to derive the authentication key $\Bbbk_{aut}$and the encryption key $\Bbbk_{conf}$. The 8-bit data fields suggested in Figure 4.3 serve us only as a baseline, giving the minimum expected size for such frames. We decided to scale down the 16 bits data field to 8 bits since this resolution should be enough for most applications and will significantly improve energy consumption. If larger data is collected, then the fields can be extended to 16-bits without impacting the authentication tag which stays at 32 bits, however this will require to switch to a larger block size for SPECK as well.

4. *Frame encryption* is added only as an option but is not part of the default protocol description as outlined in Figure 4.3 (ii). The reason behind this is that the benefit of encrypting the frames seem to be somewhat marginal. The main intention is to assure privacy for the user, e.g., the ID of the sensors is hidden and thus it cannot be tracked (this is the motivation from

Handshake for session-key establishment

1. $\mathsf{Sens_{ID}} \rightarrow \mathcal{R}$: $\mathsf{Sens_{ID}}, N_{\mathsf{Sens_{ID}}}$
2. $\mathcal{R} \rightarrow \mathsf{Sens_{ID}}$: $\{N_{\mathsf{Sens_{ID}}}, N_{\mathcal{R}}, \mathcal{R}\}_{\Bbbk_{ms}}$
3. $\mathsf{Sens_{ID}} \rightarrow \mathcal{R}$: $\{N_{\mathcal{R}}, N_{\mathsf{Sens_{ID}}}\}_{\Bbbk_{ms}}$

Figure 4.2: Example of session key establishment based on ISO 9798-2-4 three-pass mutual authentication based on symmetric primitives (as suggested in [76])

[102]). However, it is unclear if privacy should really be a concern since there are countless way to track a car besides recording the ID of the TPMS sensors, e.g., the myriad surveillance cameras dispatched on the roads. Besides tracking the user, exposing information related to tire condition to outsiders seems to have no security implication (finally, the condition of tire, in extreme cases, e.g., deflated tire, can be recognized by outsiders from visual inspection). However, if assuring confidentiality of the frame is a desired security objective, then we can enable the encryption option. The steps perfomed by the sender sensor are summarized in Algorithm 1. The check for authenticity is depicted in Algorithm 4 and proceeds from verifying the session counter and id up to checking the tag which is the more demanding operation (note that verification is performed only by receivers). Finally, Algorithm 3 outlines the steps of the receiver. Here in order to avoid an anonymization of the sensor ID or sending it in cleartext in order to select the particular key, we opted out for trying to decrypt with each key and check its authenticity. This procedure does not add significant additional costs since decryption requires only XOR-ing with a key stream that already exists (and will be used when the frame encrypted with the corresponding key arrives). Moreover, frame verification in case when the selected key is not that of the sender will likely fail at verifying the session id so it does not incur additional costs, e.g., checking for authenticity. To obtain a crisper verification algorithm, since we used encryption in counter mode, one can simply avoid encryption of the ID by using an 8-bit mask. In this case the receiver's algorithm from Algorithm 3 can simply proceed with the key allocated for a particular ID. Encryption of the counter can also raise some concerns in case that frames are lost, however since frame emission takes place at fixed intervals, e.g., 2 minutes, it is easy for the receiver to retrieve the current value of the counter by simply checking the elapsed time since the protocol started. Again, as the counter holds no critical information, it can be sent in plain-text as well (this is just a minor implementation decision).

Algorithm 1 Sender's algorithm (as proposed in [76])

```
1:  procedure BUILD AND SEND FRAMES
2:      SetSessionKey()
3:      sess_cnt←0
4:      repeat
5:          if sess_cnt>max_sess then
6:              SetSessionKey()
7:          end if
8:          data←ReadSensorData()
9:          frame←concat(data, sess_cnt)
10:         frame←concat(frame, sess_id)
11:         if opt_conf then
12:             kstream←Cipher(𝕜_conf, sess_cnt)
13:             frame←frame⊕ kstream
14:         end if
15:         tag←MAC(𝕜_aut, frame)
16:         frame←concat(frame, tag)
17:         SendFrame(frame)
18:         sess_cnt←sess_cnt+ 1
19:     until true
20: end procedure
```

Algorithm 2 Frame Verification (as proposed in [76])

```
1:  procedure CHECKFRAMEAUTHENTICITY(frame)
2:      id←extract(frame, ps_id, ln_id)
3:      sess_cnt←extract(frame, ps_cnt, ln_cnt)
4:      if sess_cnt>sess_cnt[id] then
5:          sess_id←extract(frame, ps_sid, ln_sid)
6:          if sess_id=sess_id[id] then
7:              tag←extract(frame, ps_tag, ln_tag)
8:              tag' ←MAC(𝕜_aut[id], frame)
9:              if tag = tag' then
10:                 data←extract(frame, ps_data, ln_data)
11:                 return data
12:             end if
13:         end if
14:     end if
15:     return ⊥
16: end procedure
```

Algorithm 3 Receiver's algorithm (as proposed in [76])

```
1: procedure RECEIVE AND VERIFY FRAMES
2:     repeat
3:         frame←ReceiveFrame()
4:         if opt_conf then
5:             for i ← 1, n do
6:                 frame' ←frame⊕ kstream[i]
7:                 data←CheckFrameAuthenticity(frame')
8:                 if data≠⊥ then
9:                     frame←frame'
10:                    exit for
11:                end if
12:            end for
13:        else
14:            data←CheckFrameAuthenticity(frame)
15:        end if
16:        if data≠⊥ then
17:            id←extract(frame, ps_id, ln_id)
18:            sess_cnt[id] ←sess_cnt[id] + 1
19:            kstream[id] ←Cipher(𝕜_conf[id], sess_cnt)
20:        end if
21:        data←⊥
22:    until true
23: end procedure
```

4.1.3 Overview of experimental results

For the full experimental results the reader is referred to [76], here we provide only a brief overview. The devices from our experiments are two dedicated TPMS development kits from Infineon. The SP37T development kit [40] was used for programming the TPMS sensors. It is based on a standard 8051 low performance microcontroller and includes an LF receiver, an RF transmitter unit and an ADC converter for signal conditioning. In Table 4.1 we summarize the characteristics of the SP37 sensor.

Our implementation was centred on two lightweight block cipher around which we build a CBC based MAC code. Indeed, the cryptographic primitives would be ideally implemented in hardware, but such support is absent from current controllers. Even in software, the implementation that we use in our protocol design, proved to be energy efficient with the additional costs for cryptographic computations being at only 2.77μJ which is around 20% from the total cost of a regular frame transmission at 13.11μJ. This cost can be actually nullified by a proper hardware implementation which is certainly to come for TPMS sensors. At the communication level, the size of the frame is expanded by the 32-bit authentication tag which (along with the costs of the underlying cryptography) increases consumption from 13.11μJ to 20.34μJ which is roughly 50%. We underline that this cost is in fact an upper bound since the authentication tag is equal in size with the actual sensor data (32 bit) while with a larger data-field this percentage decreases as the authentication tag remains constant in size.

Compared to the energy consumption reported by previous work [102] our result is two or-

Characteristic	Infineon's SP37 (8051 based)
Operating Voltage	3.3V
Supply voltage range	1.9 - 3.6V
Digital I/O Pins	19
Analog Input Pins	8
DC Current per I/O Pin	10 mA
Flash Memory	**6 KB**
RAM	**256 Bytes**
EEPROM	**31 byte emulated EEPROM** (+ 12 KB ROM)
Clock Speed	12 MHz
Temperature range	-40°C to 125°C

Table 4.1: Characteristics of Infineon SP37 8051 sensors (from [76])

ders of magnitude lower, this is of course mainly due to the dedicated TPMS sensors from the deployment platform but also due to the lighter cryptographic constructions, i.e., SPECK.

For more comprehensive practical results and more details on the protocol we refer the reader to the published version of our work in [76].

4.2 Smartphone based car access via NFC [2]

Smart-phones have become ubiquitous devices, they are everywhere and everybody has them. They may not be ideal as keys for cars, but they are becoming such a wide-spread device that their potential use as car keys deserves at least some atention. We must be honest in accepting that there are at least two issues with accepting smart-phones as car keys:

- *Security*, since smart-phones are host to a myriad of applications, not all of them innocuous, they are not as secure as your regular car key. Even for regular car keys, there were so many vulnerabilities reported in the past, not unexpected as the key is the entry point for cars and thus the main attack surface. Verdult et al. [96] points on at least four issues: i) lack of (or weak) pseudo-random number generators which lead to replays, ii) predictable passwords, many cars use the same key, iii) memory dump allow recovery of the full key (the protection bit is not set) iv) weak cryptography (e.g., non-AES designs). A distinct line of research, [86], reports vulnerabilities on an Atmel immobilizer protocol stack. The good news is that all the vulnerabilities can be fixed by the appropriate security designs and proper choice of the underlying cryptography. Open problems remain on more advanced security topics such as side-channel attacks, distance bounding protocols, etc. but these are not within our scope here.

- *Usability*, while some would be quick in saying that smart-phones are easy to use as access keys, in-depth studies show that it may not be always so, see [5]. But there was much scepticism in many directions of smart-phones evolution were predictions proved to be so

[2]The results presented in this section are based on author's previous research presentation at an industry conference: Adrian Radu (Continental Automotives), Bogdan Groza (UPT), *Security Concept for Smartphone Car access via NFC RF ID Device*, Continental Software Conference, Regensburg, 2015.

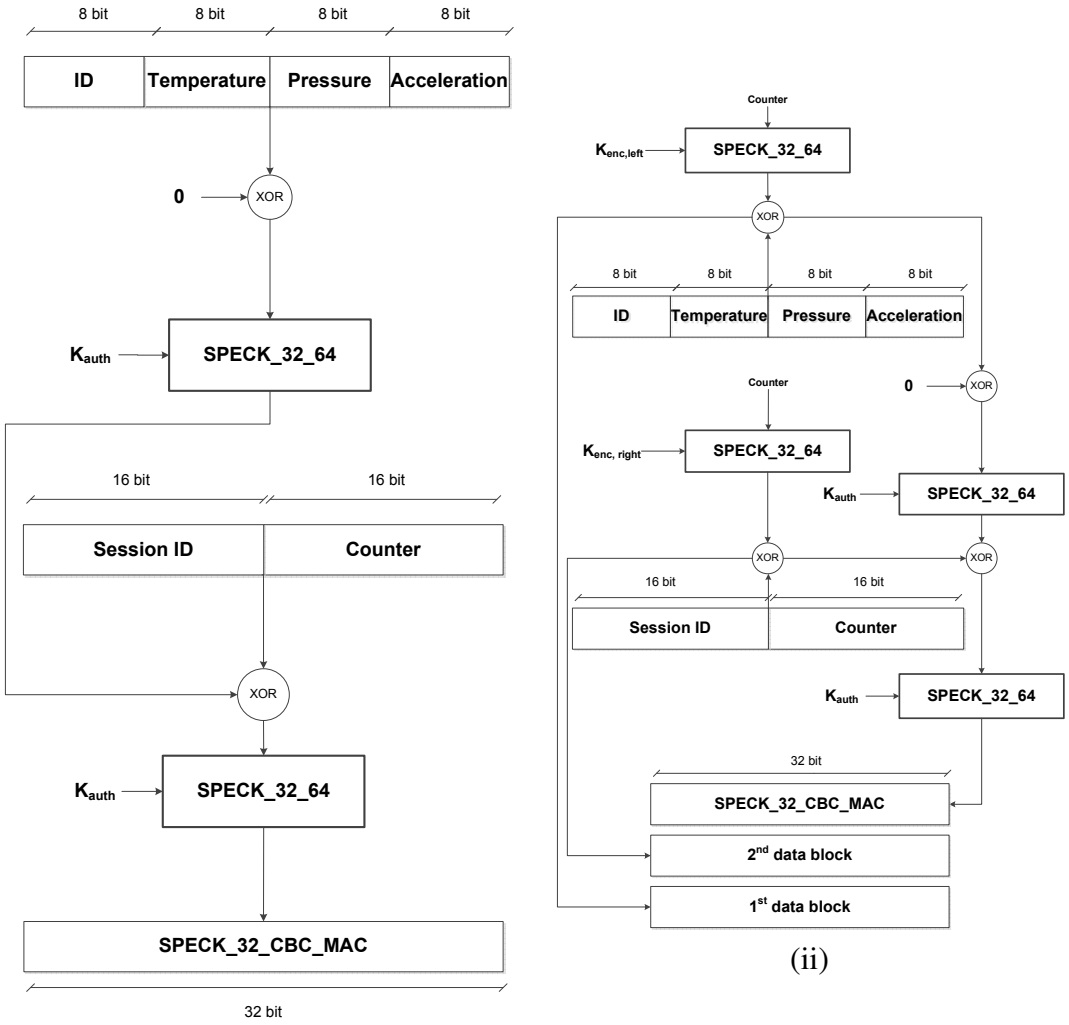


Figure 4.3: A 96 bit frame: i) authenticated with CBC-MAC and ii) with authenticated encryption (encrypt-then-MAC) based on SPECK CBC-MAC (as proposed in [76])

wrong (e.g., there are so many cons on the usability of big screen smart-phones and yet these are so wide-spread and used).

This was the downside, the advantages in using smart-phones as car keys are numerous, we just highlight two of them:

- *High flexibility in terms of user interfaces & functionalities*, one can add any functionality to the key, e.g., open any door or window, display information about the car, etc., as the interface is not bind by a particular physical design (as in the case of regular keys),
- *Remote configuration* of the key, e.g., via the cloud, delegate access rights to other keys (phones) or even locate your lost key (GPS) or remotely kill it once it is lost.

Here we will only briefly describe a design that was taken into consideration by Continental Corporation and where we were invited to contribute [68].

4.2.1 System overview and protocol design

We discuss some of the security objectives for the application. Given that this project is still under development and that certain parts will likely be confidential due to the policy of the industrial partner that is involved, we only state some of the objectives as they were presented in [68]:

1. The *registration* protocol suggested in Figure 4.5 is responsible for pairing a mobile phone with an NFC key, this means establishing a shared secret key between the phone and the token. Our initial choice was to rely on a visual channel by using a master reset key stored as a data matrix on the cover of the key. This of course requires safe deposit of the token cover and may bear some similarities to the to the PIN-PUK pair in mobile phones were the PUK (PIN Unlock Key) is a longer key required to set the smaller PIN (Personal Identification Number). The visual channel is used in order to avoid input from a virtual keyboard which is a more demanding procedure and requires minimal user input (just open the application stored on the phone).

2. *Token-phone mutual authentication* is to be performed by standard challenge-response authentication. One of the design intentions is to base as much as possible on existing standards, e.g., ISO/IEC 9798-2 compliant authentication. The design in Figure 4.5 is only a starting point.

3. The *access control policy* is responsible for granting particular functionalities to each user (phone). This is based on an Access Control List (ACL) with a cryptographic integrity check that is stored inside the token. The list is also augmented by the lifetime of the rights (i.e., number of attempts) and establishes if the user can delegate a part of high rights further. Haptic feedback is a response from any of the user's action to the token to eliminate (at least in part) jamming attacks by making the user aware if the function was correctly received by the car.

4. *Rights delegation* allows to transfer a subset of the rights from one device to another. Several options are considered: delegation lifetime, fixed number of access requests, time or location dependency, etc.

5. *Revocation* allows un-pairing a phone from the token. Any phone can revoke its connection to a key by simply erasing its shared key, but the master key on the cover allows one to revoke any other phone that was already paired with the token. Additionally a user master-reset password can be set for this purpose. Implementing a kill-switch that simply erases the key from the token from the remote is a desired functionality (e.g., in case of theft).

6. Other functionalities: *remote editing* of the access rights is also a planned functionality. *Traceability* which allows to recover the full history for diagnostic purposes is also considered.

For obvious reasons, the industry cannot be so fast in coming up with this NFC key as it would be to turn it into a conference paper. This project is ongoing at Continental and is subject of continuous changes and improvements.

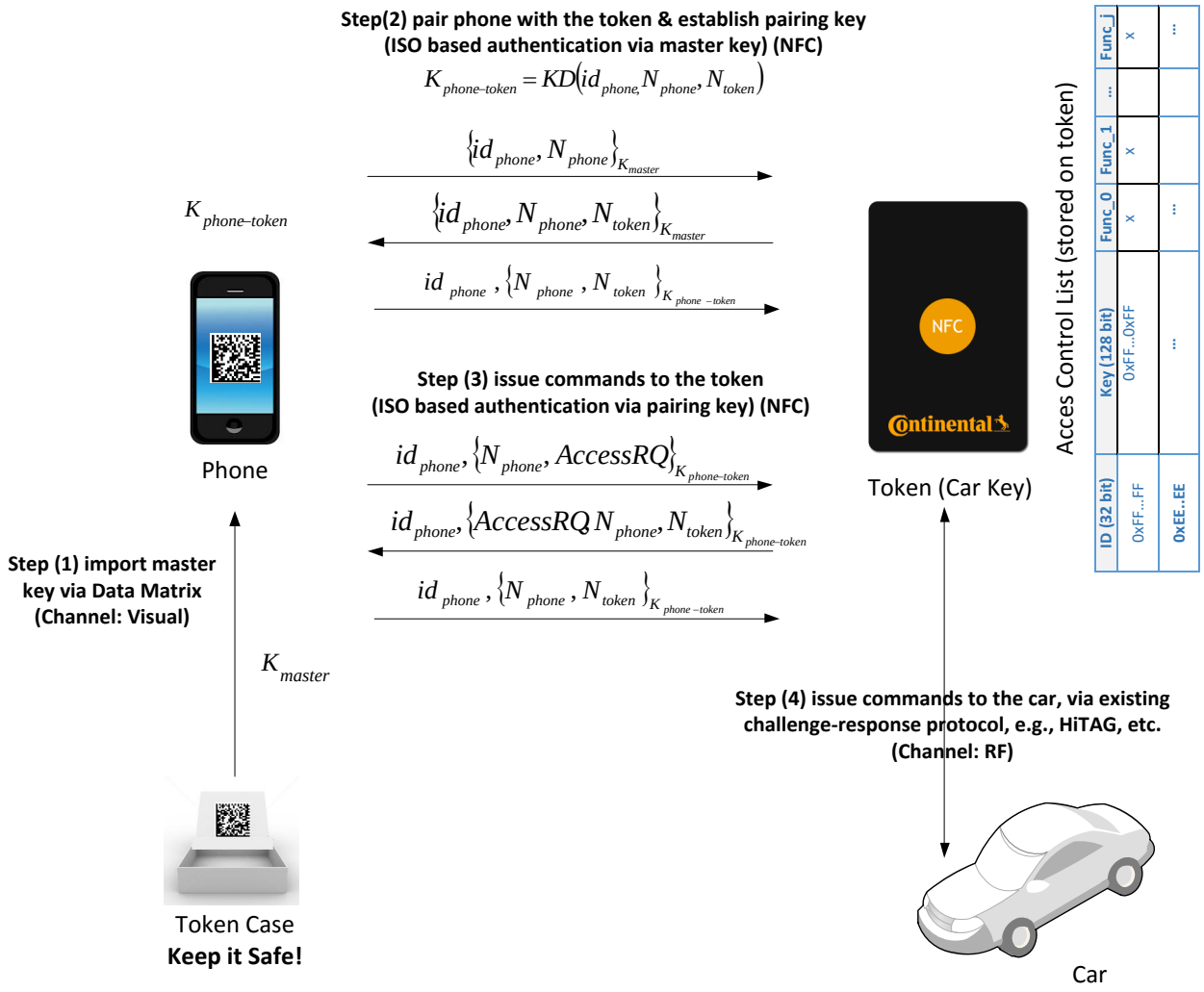


Figure 4.4: Suggested handshake for pairing an NFC token with a smartphone (based on [68])

4.2.2 Randomness, a relapsing issue [3]

We are all aware that good quality randomness is mandatory for security critical tasks such as generating encryptions or authentications keys. Breaking cryptography is easy if these keys are predictable. It is somewhat surprising how this problem has so many solutions and yet it relapses in practice so many times. We include the exposition of randomness generation here since it is very likely that some of this work will be used in the design of the wireless keys. The design of this

[3]The results presented in this section are based on author's research report: George Tipa (Continental Automotives), Bogdan Groza (UPT), *High quality randomness for safety critical tasks on automotive embedded devices*, that stayed at the core of a patent submission by Continental Corporation: European Patent Application, EP 14465511.5-1953/28.05.14, 2014

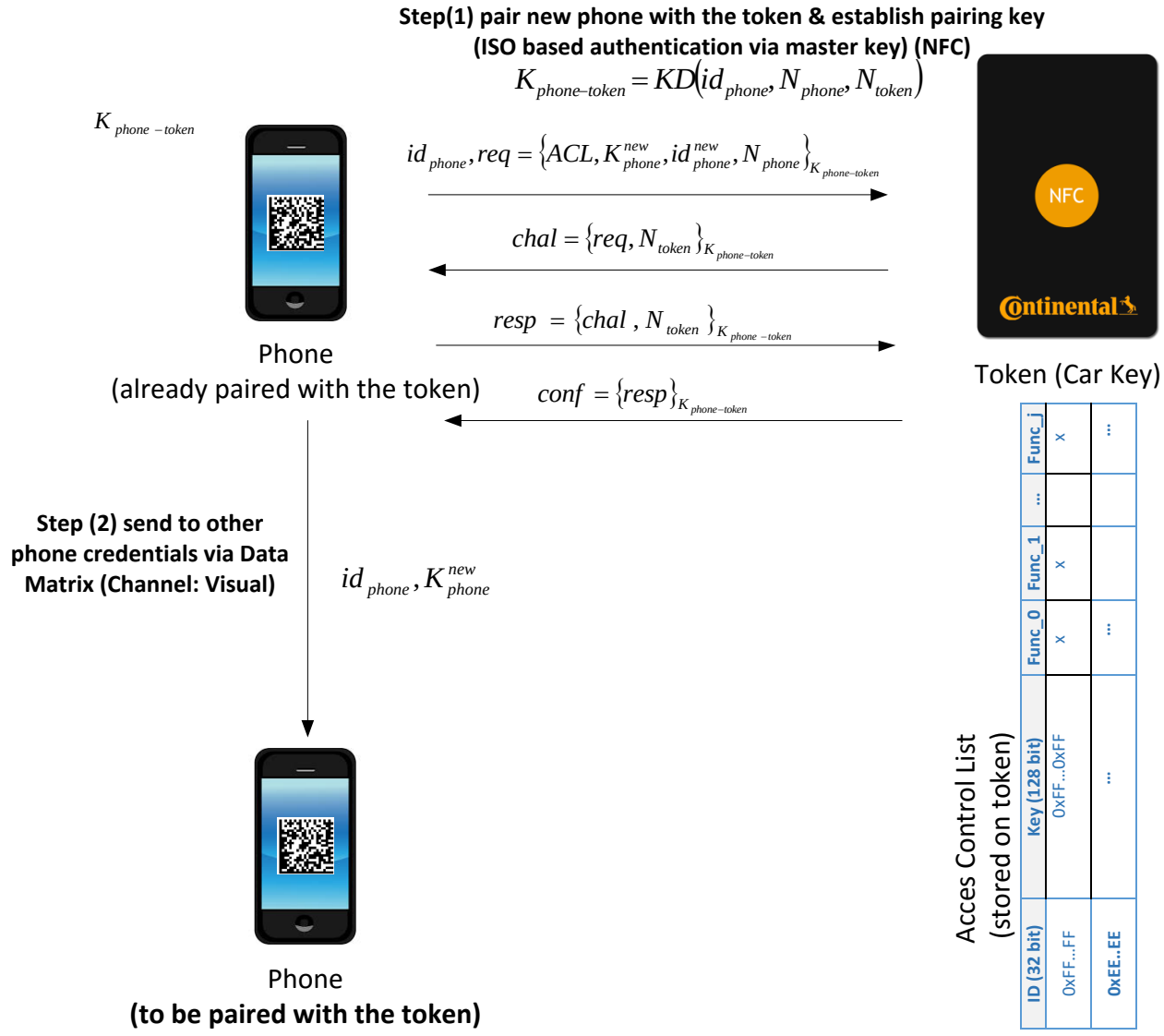


Figure 4.5: Suggested handshake for rights delegation to another smartphone (based on [68])

procedure started as a joint work with George Tipa (Continental Corportation) and later a patent submission was made by Continental Corporation [88] based our research [87]. The exposition from this section is mostly based on our research report [87].

True random numbers generators (TRNG) are produced by exploiting system characteristics that are hard (or nearly impossible) to predict and there are many hard to control properties of wide-spread hardware that are commonly employed in practice: the noise from Zenner diodes collected by the ADC or the measured drifts of two oscillators. But not all devices are yet equipped with these peripherals and for cost reduction one does not want to add extra circuitry. More recently, it was proved that the state of SRAM after start-up is a good source for randomness [37], this is a better alternative since a large number of devices are equipped with SRAM. The finding was confirmed by subsequent research [93] and one limitation proved for this methodology was that it may not be reliable at low temperatures, e.g., -30° Celsius.

This is in contrast with pseudo-random number generators (PRNG) which are using computational methods, e.g., one-way functions, to generated unpredictable sequences from a seed. This alternative is not preferred due the difficulty in managing the seed which must be random, or at least unique for each new re-initialization (the PRNG produces the same sequence if the initialization is done with the same seed).

OUR PROPOSAL. The mechanism was designed to be able to gather sufficient entropy in a short amount of time as waiting for devices that slowly collect entropy is not an option. We tried to garner entropy from any existent device on the board (adding extra-circuitry such as a Zenner diode, while efficient, is not an option) and we reached the conclusion that there are 5 hardware units that can generate certain amounts of entropy, we describe these next. The uninitialized *Random Access Memory (RAM)* contains bits with hard to predict states when the controller switches from a low-power state or power is applied for the first time because of uncontrollable manufacturing conditions (memory cells have a bi-stable behaviour). The *Analog to Digital Converter (ADC)* is a rather classical unit for generating randomness, unfortunately we did not have a specialized circuitry linked to it so we could at best count on burst measurements from which only the LSB is considered (hopefully, this gathers measurements errors). The *Clock Calibration Unit (CCU)* provides possibilities to calibrate the internal RC oscillator with respect to an external oscillator while the result of this calibration represents the number of periods of the external oscillator counted inside the period of the internal oscillator - this complies with measuring oscillator drifts, a common procedure for generating randomness. The *Bus Performance Counter (BPC)* measures the number of write accesses to the on-chip bus (all write accesses from the CPU and DMA bus masters to peripherals, registers, external BUS and other sources are measured). Finally, the *Timer* can be initialized at start-up and configured to use the maximum time base, the exact value can be unpredictable after a while.

Merging all these sources of entropy makes sense since each of them has individual drawbacks while merging does not lead to delays or computational overloads since the procedure is not triggered very often. We enumerate some shortcomings in what follows. The RAM may have failures if the controller operates at low temperatures as shown in [93]. Even if the controllers do not usually work at extremely low temperatures, hackers can induce such temperatures in the circuitry to prevent good quality randomness and bypass underlying security mechanisms, e.g., cryptographic authentication. The ADC may fail to provide appropriate randomness if the input pin is connected to the ground, or even if this is not the case if the input pin is available to an adversary it can perform measurements of his own. The Clock Calibration Unit is only a very slow source of entropy since oscillator drifts are expected in the order of only several parts per million (ppm). Finally, the Timer and BPC are not a good source of entropy since their values can be predicted under certain conditions (for example, shortly after start-up, they do not increase considerably).

Table 4.2 provides an overview of potential automotive target platforms and their capabilities, it is only the BPC that may be missing from some of the platforms. For the experiments performed at Continental Corporation, Spansion FR81 is the platform that was used. This platform has all the aforementioned hardware resources.

OVERVIEW OF THE SCHEME. In this exposition we skip the low level details of our scheme. From an upper view, the scheme is based on two stages. First is the *start-up stage* which collects entropy out of the uninitialized SRAM. This stage uses an SRAM-Selector that uniformly selects memory locations in order to build a vector of predefined entropy and then passes it through a Mixer which applies a one-way function over the input to return an output of fixed length. This stage is based on an algorithm that we call SRAM-SelectorMixer which will be detailed below.

Producer	Device	ADC	Timer	SRAM	BPC	CCU/CMU
Spansion	**FR81**	**x**	**x**	**x**	**x**	**x**
Spansion	FR60	x	x	x		x
Renesas	V850	x	x	x		x
Freescale	MPC560	x	x	x		x

Table 4.2: Target platforms and their capabilities (based on [87])

Second, is the *request stage* which collects entropy whenever a random value is requested from the ADC, CCU, BPC and Timer. These are all concatenated with the output of the start-up stage and feed as input to a OWF.

BLOCK DIAGRAM OF SRAM-SELECTORMIXER. A block diagram of the Algorithm may be helpful for an overview. The algorithm jumps at random positions within the memory and selects memory bytes to build up a vector that will be subsequently compressed to return the random value. Figure 4.6 gives a block overview over the selector-mixer and Figure 4.7 gives a suggestion of how it works. A precise description of the algorithm is given below. The next byte to be visited is chosen based on a predefined seed and the value from the current memory location. This way, it ensures that during each start-up phase different memory locations will be selected. The seeding value will be written in a specific location within the EEPROM during the production phase (moreover, the seed can be different in case of each controller).

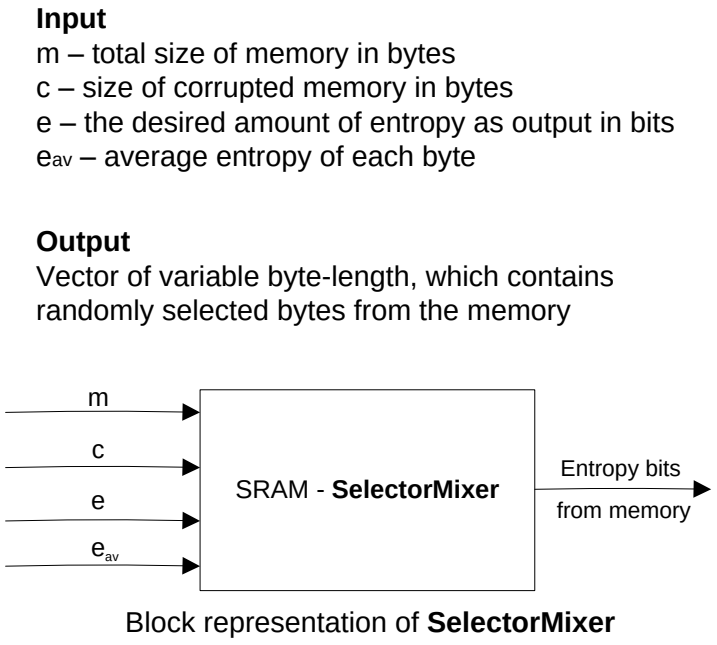


Figure 4.6: Block representation of the selector-mixer component (based on [87])

A MATHEMATICAL TREATMENT OF THE SRAM-SELECTORMIXER. This algorithm is re-

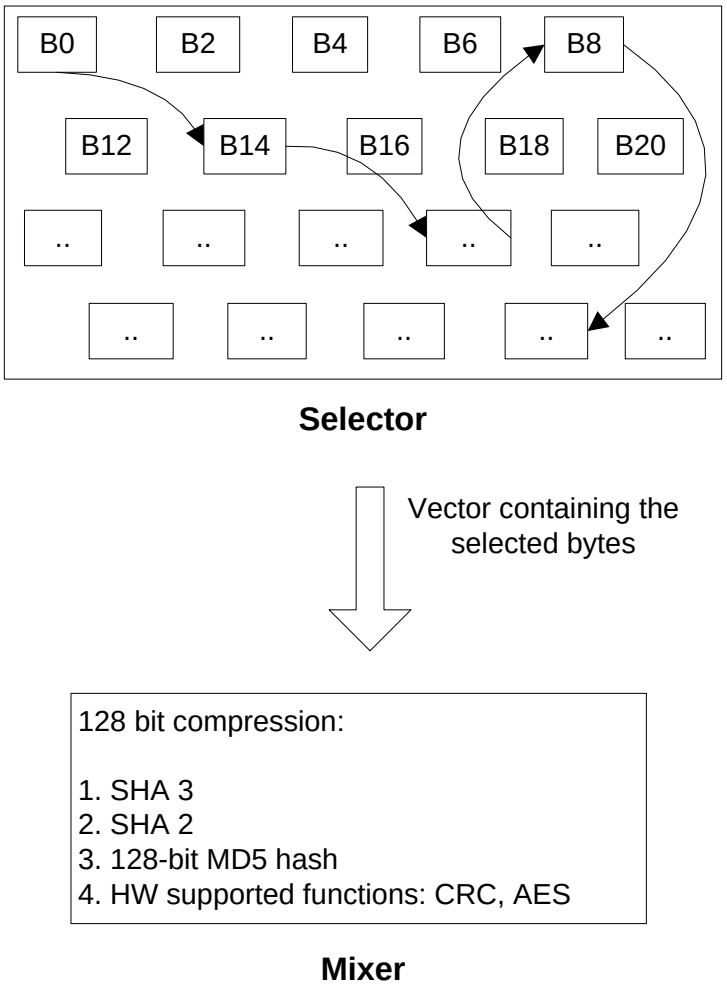


Figure 4.7: Suggestion for the selector-mixer procedure (based on [87])

sponsible for the gathering entropy out of the memory. Based on fixed input parameters it will output a vector of bytes containing the requested entropy. We assume that the entropy is uniformly distributed over the entire memory, except for specific areas, referred as corrupted memory (in our practical setup this corresponds to memory that is pre-loaded with fixed values, e.g., the OS stack). The bits are extracted by walking randomly over the memory. To bring more details on the scheme, we first need to fix some terminology.

We employ the classical definition from [75] to define the **guessing probability** of a byte b as the maximum probability over the experimental data that is available:

$$\gamma(\mathrm{b}) = \max\{\Pr[\mathrm{b} = g] : g \in 0..255\}.$$

Note that in case of uniform distribution this is 2^{-8} while in our case this is computed over the experimental data. The **minimum entropy** of one byte is the base 2 logarithm from the inverse of the guessing probability. Consequently, the minimum entropy of a piece of memory $\mathbb{MEM}$ is the

base 2 logarithm of the inverse of the product of guessing probabilities for all bytes in the memory:

$$e_{\min}(\mathbb{MEM}) = \log_2 \left(\prod_{\forall \mathrm{b} \in \mathbb{MEM}} \gamma(\mathrm{b}) \right)^{-1}.$$

Assuming uniform distribution of the entropy for each byte we define the **average entropy per byte** as the minimum entropy of the memory divided by the size of the memory, i.e., the number of bytes:

$$e_{av} = \frac{e_{\min}(\mathbb{MEM})}{\textbf{size}(\mathbb{MEM})}.$$

ALGORITHM SRAM-SELECTORMIXER. The algorithm for selection and mixing SRAM-SMix takes as input: the total size of memory m (in bytes), the size of the corrupted memory c (in bytes), the target entropy e (in bits) and the average entropy of each byte e_{av} entropy (bits/byte). The size of the corrupted memory c and the average entropy of each byte e_{av} are determined by experimental results. The algorithm uses a master key K_{master} which is a non secret random value that is given as seed to a one-way function that outputs 16 bits that are used as pointer to the next memory location. The intention of this algorithm is to select n memory bytes uniformly distributed in memory such that the total entropy of the bytes is e, these are further feed as input to a one-way function in order to create the random value r that is the output of the algorithm.

Algorithm 4 Randmoness Extraction from SRAM (based on [87])

procedure SRAM-SMIXER(e, e_{av}, m, c)
  $n \leftarrow \mathit{toint16}\left(e \cdot \mathrm{m} \cdot [e_{av} \cdot (\mathrm{m} - c)]^{-1}\right)$
  $v \leftarrow \mathit{malloc}(n)$
  $i \leftarrow 1$
  $k \leftarrow 1$
  while $i \leq n$ **do**
    $k \leftarrow \mathrm{OWF}_{16}(i||K_{master}||k||\mathbb{MEM}[k])$
    $v[i] \leftarrow \mathbb{MEM}[k]$
    $i \leftarrow i + 1$
  end while
  $r \leftarrow \mathrm{OWF}_{128}(v)$
  return r
end procedure

Proof. We prove that algorithm SRAM-SMixer selects a vector v of total minimum entropy e. Assuming uniform distribution from the output of OWF_{16} the probability the j bytes are selected in the corrupted memory area is:

$$A = \Pr[\mathbf{Corr}(j)] = \binom{n}{j} \frac{(\mathrm{m} - c)^{n-j} c^j}{\mathrm{m}^n}$$

Consequently, the expected number of corrupted bytes that are chosen is:

$$\sum_{j=0,n} jA =$$

$$= \sum_{j=0,n} j\binom{n}{j}\frac{(\mathrm{m}-c)^{n-j}c^j}{\mathrm{m}^n} =$$

$$= \left(1-\frac{c}{\mathrm{m}}\right)^n \sum_{j=0,n} j\binom{n}{j}\left(\frac{c}{\mathrm{m}-c}\right)^j =$$

$$= \left(1-\frac{c}{\mathrm{m}}\right)^n \left(\frac{c}{\mathrm{m}-c}\right) \sum_{j=0,n} j\binom{n}{j}\left(\frac{c}{\mathrm{m}-c}\right)^{j-1} =$$

$$= \left(1-\frac{c}{\mathrm{m}}\right)^n \left(\frac{c}{\mathrm{m}-c}\right) \frac{d\left[\sum_{j=0,n}\binom{n}{j}x^j\right]}{dx}|_{x=\frac{c}{\mathrm{m}-c}} =$$

$$= \left(1-\frac{c}{\mathrm{m}}\right)^n \left(\frac{c}{\mathrm{m}-c}\right) \frac{d(x+1)^n}{dx}|_{x=\frac{c}{\mathrm{m}-c}} =$$

$$= \left(1-\frac{c}{\mathrm{m}}\right)^n \left(\frac{c}{\mathrm{m}-c}\right) n\left(\frac{c}{\mathrm{m}-c}+1\right)^{n-1} =$$

$$= n \cdot c \cdot \mathrm{m}^{-1}$$

This leads to an average total entropy of $e_{av} \cdot n \cdot (1 - c \cdot \mathrm{m}^{-1})$ and by replacing n with $e \cdot \mathrm{m} \cdot [e_{av} \cdot (\mathrm{m} - c)]^{-1}$ this gives a total minimum entropy of e.

EXTRACTION TIME. Algorithm SRAM-Selector is triggered whenever the controller enters in a normal run state, following either a low-power state or a reset event. In instrumentation clusters or body controllers, this occurs whenever the car is stopped and the ignition is unplugged. During the low-power state, the controller is configured to cut off the RAM power supply in order to save power. Consequently, the memory cells will have undefined values on return from the low-power state. In figure 4.8 we outline the transitions between the three states: reset-state, normal run and low-power state. The reset state occurs when the controller is powered on for the first time. The normal run state occurs after leaving the low power mode or after reset (in this state the controller is fully functional and the RAM is powered up). The low-power state occurs to save power if the controller is no longer used (the RAM power supply is cut off, after power-on the bits will be in an undefined state).

PREDICTED VS. MEASURED MINIMUM ENTROPY. Figure 4.9 shows the guessing probability for 65535 bytes of memory. Note the entropy gap after the first 6 kilobytes of memory since this area is already initialized by the OS. The minimum entropy of the entire memory (65535 bytes) is of 34141 bits, resulting in $e_{av} \approx 0.52$. Figure 4.10 shows plots of the predicted minimum entropy compared to the minimum entropy of the bytes selected at random from memory.

TEMPERATURE INFLUENCE ON ENTROPY. The guessing probability was computed for each byte out of the 65535 bytes of memory after 200 low-power states at four temperatures: $-20°$ Celsius, $0°$ Celsius, $20°$ Celsius and $20°$ Celsius. Figures 4.11 depicts this for $-20°$ Celsius and by

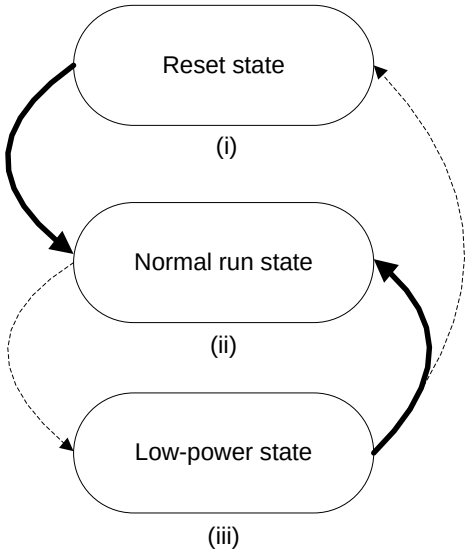


Figure 4.8: The state transitions (based on [87])

visual inspection there were no notable differences between any of the measurements. Then in Figure 4.12 we compare these values to show the temperature influence on the variation of the entropy (by mathematical computation). Results are depicted for -20° Celsius (blue), 0° Celsius (green), 20° Celsius (red) and 40° Celsius (magenta). We note that entropy variations are not significant, though the best results are apparently at 0° Celsius (green). Similar results were achieved for a 2nd board on which we performed the same tests. All of the memory dumps were done by Continental employees and our contribution was on the design of the algorithm and analysis/interpretation of these results.

STATISTICAL TESTS. The results were subject to extensive tests based on the battery of tests proposed by NIST [73]. All tests were successfully passed, but we all know that these are only statistical tests and provide only a rough indication for the security of RNGs.

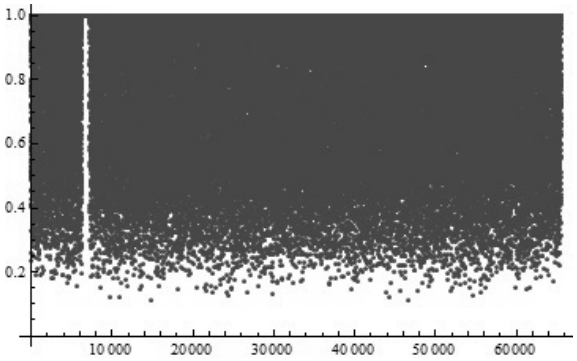


Figure 4.9: Guessing probability for each of the 65535 bytes of a memory sector (based on memory dumps received from Continental [87])

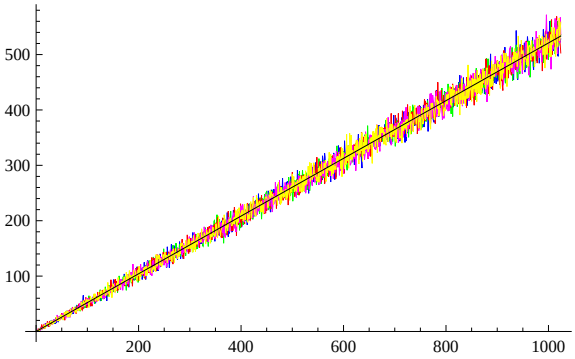


Figure 4.10: Predicted minimum entropy (black) vs. minimum entropy of randomly selected bytes (color) (based on memory dumps received from Continental [87])

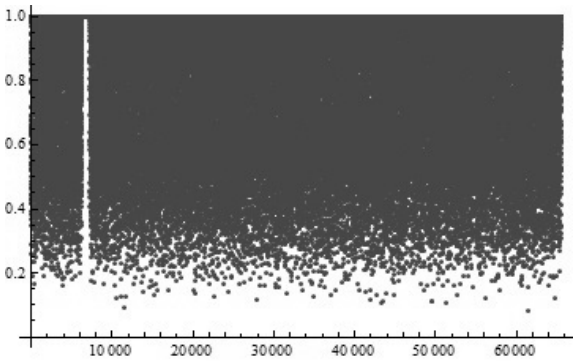


Figure 4.11: Guessing probability for each of the 65536 bytes of memory at $-20°C$ (based on memory dumps received from Continental [87])

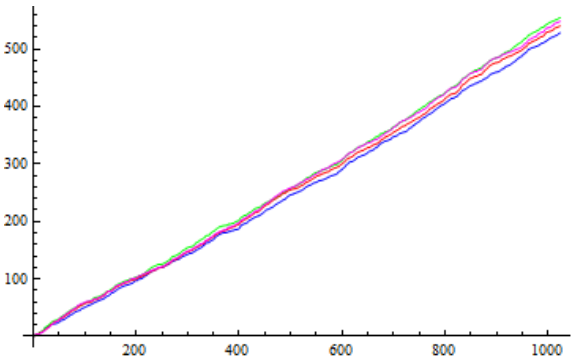


Figure 4.12: Increase of minimum entropy for the first 1024 bytes at $-20°C$ (blue), $0°C$ (green), $20°C$ (red), $40°C$ (magenta) on 1st board (based on memory dumps received from Continental [87])

Part II

Future plans

Chapter 5

Challenges and future developments

In this chapter I will enumerate some future development plans based on research directions that I consider to be open and relevant for automotive security. Niels Bohr is sometimes credited for the following (humorous) remark: "Prediction is very difficult, especially about the future". For this reason let us be moderate in anticipating the future of cars, a decade ago it would have been sufficient to have a car that runs cheap on fuel, nowadays one wants a hybrid electrical vehicle that plays his favourite songs, automatically adjusts the ambient by recognizing the driver, instructs him on weather conditions, drives itself, etc. – and all these, cheap.

5.1 Education and research at UPT

Education is a key issue not only for producing capable engineers that will known how to tackle future security problems, but also in creating responsible users that are core to the security of the system. Since human factors are the main source for various attacks, e.g., by social engineering, the implications can go far beyond.

Undoubtedly, here at UPT we made significant progress in developing our educational tools for future engineers in the field of automotive systems. We have laboratories and equipments that are industry standard and we benefit of good interaction with the industry (though our research collaboration with the industry is still needy and we need more progress on that). I will not waste space for describing our infrastructure here. Of course, there is always more to be done for the research infrastructure, however, our shortcoming now is on the human resources required for high quality education and research. We are short in terms of employees and this is because of the poor and uncertain funding.

To alleviate most of the problems that we face, we need more research grants. Relying on the industry for funding is a debatable option since this unavoidable turns the university professor or researcher into an employee of the company and finally nullifies the role of the university. The interests of the industry are antagonistic with that of universities in many matters. In Table 5.1 we try to make a comparison in terms of research interests on the university vs. industry's side. This is of course a subjective matter, but we can accept that generally university research is targeted on subjects that are easy to sell on the scientific market and get easily cited and used by others, these are rarely saleable as products on the market. In contrast, the industry seeks for profit, generally in the short term, and tries to protect its ideas such that others are unable to use them unless they pay, sometimes much more than the idea actually worths. This is an uneasy subject that we cannot solve here, we just state that we are aware of the problem.

Academic research requirements	Industry research requirements
marketable as scientific paper (**idea**)	marketable as product (**device**)
open doors for future research (**citable**)	improve on day-by-day life (**usable**)
want others to use the idea for free (**get cited**)	do not want other to use the idea for free **(patents)**
brings profit (if) on the long term (e.g., **decades**)	profit in the short run (e.g., **years**)

Table 5.1: Potential differences between academic and industry research requirements

Our future preoccupations should be focused on having even stronger relations with the industry and apply for common research projects that can mediate between our interests when they are antagonistic.

5.2 Formal proofs of security

Rigorous security relies on rigorous proofs. The are two distinct ways to address security proofs. On one hand we have handwritten proofs for protocols and cryptographic primitives which can benefit from the ingenuity of the solver but can also be subject to human error. On the other, we have automated proofs generated based on a formal description of the protocol or cryptographic primitives. This usually results in a longer proof, harder to read and not necessarily tight in the security bound, but has the guarantee of being correct. There are both pros and cons for each of these approaches. Clearly, without a strong evidence for security, i.e., a proof, a proposal is insufficient for practice.

There were many concerns in the recent years build around security proofs for cryptographic primitives, such proofs are hard to read and hold errors that are left undiscovered for years. However, this may not be such a big concern since most of the time these errors were fixable, at least for most primitives that are now of practical relevance. In fact, most of the errors are in papers that never get to be used in practice, while as soon as the construction becomes of interest, errors are found and later fixed (e.g., the famous case of OAEP secure under the RSA assumption: proved in '95 [7], shown to be in fact wrongly proved 5 years later [74] and then quickly fixed [23]).

Formal methods for protocol analysis had significant impact in the late 90s and in the decade that followed. Meadows and Lowe were pioneers of this technique [56], [51]. Today it may be that cryptographic protocols are only a marginal target of such tools since there are well studied, standardized protocols for various tasks, e.g., key-exchange, etc., while the formal verification tools may be more sucessful with rather immature proposals.

What appears to be a fruitful target for such tools is the system as whole. Automotive systems incorporate everything, from user interaction, communication interfaces, electronic components, control systems, etc. It may be a significant future challenge to be able to address such systems as whole by formal verification and not just small protocol fragments. Indeed, so far formal verification tools have dealt mostly with rather small protocols or systems models when compared to the complex succession of tasks that take place in real-wolrd cyber-physical systems, e.g., vehicles.

Pursuing our research in the area of formal proofs will be unavoidable, especially if some of the

solutions that we deploy will get some practical use. It is a far greater responsibility to launch these solution to users than to put them in an academic paper that gets accepted at some conference.

5.3 Resilience to side-channels

This subject has been well explored by the cryptologic community for more than a decade. Yet, it is still far from being solved. Whenever an adversary has physical access to a security device, the wonderful mathematical proofs of security are not enough.

Side-channel attacks are a serious threat and for automotive security, they are likely one of the most dangerous future attack. The good news is that likely the countermeasures will not be different from what has already been envisioned on other systems, e.g., smart-cards. So far, there is not much about side-channel attacks for automotive components, but the reason for this is that most attacks reported so far in vehicles were easy to mount without any side-channel.

The mere nature of cars, as objects that are left unattended on the street or parking lots, or at some unknown garage in case of troubles, exposes them to adversaries that have physical access to the components. It is likely that physical countermeasures for accessing components in the car may tackle the problem in a simpler way. However, all these mechanisms are significantly increasing the production costs. Given that manufacturers usually have low cost margins, it is very likely that in-vehicle components will remain exposed to potential adversaries for a long time.

Perhaps production cost would be the greatest barrier in adopting side-channel security for vehicle components. Just to get an image of how production costs may affect manufacturer decision (or if cost is not the cause in the following example then what can be accuse? lack of information?), it is staggering to read the conclusion of a now famous paper addressing key counterfeiting [96]: while a better chip for car keys that uses AES was less than 1$, cars worth more may thousands were still using deprecated (improperly reviewed) cryptography. Indeed, it is not clear if it was only about cost, but adoption by the industry seems always to be slower when adding even an extra cent.

5.4 Challenges with key distribution

This is an issue that is seldom trivial to fix on a research paper but extremely hard to be solved in practice. Of course, all protocols rely on some secret key. Whenever one writes a paper it assumes that there is some central system/entity that facilitates key sharing, or some certificates signed by a trusted party exist. This is a reasonable assumption for a research paper since without it one cannot go further (all papers would got stuck before a protocol description is written).

But in practice, automotive systems seem to particular raise challenges in this respect. Components inside a vehicle originate from dozens of manufacturers. Some of them were designed well after the vehicle itself. It is impossible to rely on some secret keys that are embedded by the manufacturers inside the vehicle. The obvious choice would be to rely on Public Key Infrastructure and Public Key Certificates. This appears to be the only reasonable future alternative, but it will require all components inside the cars to be PKI aware, an option which again will impact costs. The PKI has succeed in assuring security over the Internet, but it has likely failed in assuring security in many other areas were research papers predicted its success such as the plethora of applications were they replace hand-written signatures, e.g., users that digitally sign their e-mails, doctors that sign their prescriptions, etc.

Given the current image, it is likely that the PKI will turn as the only viable solution for the automotive industry. But there are clearly many challenges. One of them is the particular long term use of components inside the car which is somehow incompatible with the rather short-term validity of security software which needs to be frequently updated and fixed for bugs. Software evolution in the recent years is not very helpful as it turns out that all devices (phones, tablets, notebooks, computers, etc.) need to be patched on a regular basis. But there are many reasons for which a car owner may not want to receive updates. E.g., assume that the manufacturer of your car has been recently acquired by a company located inside an adversarial country. Would you still want to receive updates given the possibility that these are back-doored for the case of some future cyber-conflicts? Can users be forced to receive updates for a product that they own?

These issues will not be easy to clarify and since keys stay at the core of all security protocols, it is clear that key distribution will be a major issue for cyber-physical systems. In the mean time, cryptographers and securitists will publish papers assuming that key distribution is a solved issue.

5.5 Challenges related to freshness

It is rather spectacular how this, apparently trivial aspect, is in fact un-addressed (or wrongly addressed) by most protocols published in the literature.

Replay attacks are trivial attacks that consists in re-sending the same message several times. In theory we address this easily by using nonces and challenge-response, but challenge-response is not suitable for the broadcast nature of buses inside cars. Then we address this on theory with timestamps and add them to the authentication tag to indicate the time at which the message was sent. But this timestamp relies on the existence of a common clock and a common clock relies on synchronization. But then synchronization comes with synchronization errors (not to mention that in order to be secure, synchronization requires cryptography which is not cheap in terms of computational load and bandwidth). Further, relying on timestamps may open door for the reply of messages in the time frame generated by the synchronization error, or at least to be delayed with the synchronization error. To the best of our knowledge, there are no comprehensive studies so far on how would this impact the safety of the vehicle. From our own experience, given that the achievable synchronization errors should be in the order of milliseconds and vehicle control systems may be tested in a laboratory setup without considering replay attacks, we believe that this issue may lead to significant problems in the future.

In the following paragraph we revive here some notes from one of our previous works [27]. Freshness turns to be quite problematic in formal models for protocol security. Freshness in security protocols has been interpreted mostly in the traditional sense of preventing replay attacks (i.e., not accepting a message twice). Cryptography provides the proper tools to achieve this: nonces, counters and timestamps, carefully embedded in the protocol can all be used to assure ordering (establishing the order in which messages are issued) or limit lifespan (decide if a particular message is still valid). But formalizing attacks is more problematic. Syverson [81] presents a taxonomy of replay attacks in which the classical sense is extended to more than the mere replay of a message. The taxonomy considers not only classic replays, but also attacks in which a message (or part of it) is used in a context for which it was not intended, e.g., interleaving, reflection or deflection attacks. This taxonomy evades the traditional interpretation of replay attacks as used by BAN logic [12]. But in general, interleavings, reflections or deflections are treated as violations of non-injective agreement [52] and they are not necessarily related to freshness. Besides Syverson's taxonomy [81] there is little interest to use freshness for more than uniqueness. Efforts to model

time-sensitive security goals exist [19] but appear to be rather isolated. While all tools employed for automatic formal protocol verification are able to detect replay attacks, to the best of our knowledge there is no support to check for time-sensitive security goals.

In contrast to the limited interest in modelling freshness for protocol security, as underlined in the previous paragraph from our work [27], for control systems and thus automotive security this objective is likely critical. Both the security community (cryptographers) and the control systems community has tools to deal with delays, the challenge is in putting them together and modelling such protocols.

5.6 Security implications of future automotive paradigms

Every new promising paradigm that hits the market comes with its economical benefits and security issues (which are hopefully addressable). For the moment I think that there three disruptive (in a positive sense) technologies that are hitting the automotive markets and a forth which I regard as a trend rather than a technology by itself:

1. *vehicle-to-vehicle (or infrastructure) communications* which will bring connectivity to the automotive world,
2. *electrical engines (hybrid or not)* which can drastically save on fuel (and pollution),
3. *self-driving cars* which can turn our driving experience from stress and time waste to tranquil or productive moments.
4. *an over-abundance of functionalities* seem to be the recent trend in the car industry, starting from sensors, rear cameras, dynamically adjustable mirrors, phone-based access control, cloud-based support, etc.

More than a decade ago, vehicle-to-vehicle communication became a topic of high interest. It is clear that the impact of such technology could be highly beneficial, starting from route optimization to collision avoidance, there are many functionalities that can greatly benefit from the existence of communication channels between vehicles. But inevitably, this opens road for attacks and there are already hundreds of research papers that address such issues. However, the true challenge comes once such systems are largely deployed in practice. Imagine how all the research on Internet security would look, in the absence of the physical (real-world) Internet. Clearly, some of the aspects that are solved on paper will prove to be unsolved once the solution is ported to practice (or even prove not to have a solution at all). In practice, there are hundreds of implementation mistakes which open back-doors. It is likely that the practical deployment of such technology will open many research opportunities in the future, so we will hear more and more on this in the forthcoming years. It is however somewhat surprising that this technology hits the market so slow.

In the recent years, hybrid electrical vehicles became very popular. One would say that there are no security implications if the vehicle runs on petrol or batteries. However, this may not be so. A car that is cheaper on fuel is also likely to travel more. This is common sense in economics, it the price of fuel is lower, people will likely tend to buy more satisfying needs that they could not afford in the past. A car that travels more, leads to a heavier interconnected world, this bring more security implications. It don't think that this would bring drastic changes, but it is a paradigm that should not be neglected in the security landscape.

Finally, an entirely new era seems to be opened by self-driving cars. This new technology, pioneered by Google with much noise and silently by many automotive manufacturers, has the potential to drastically change the landscape. But there are little doubts that for vehicles that can be stirred from the electronic brain, security will play a critical role as any back-door inside the software has the potential to turn the vehicle into a weapon. Open questions related to self-driving cars are numerous. Starting from who is responsible for the accidents (the one who designed the brain of the car vs. the driver or owner of the car) to what are the legal privileges of authorities against such cars (can the police request a self-driving car to stop or force it to follow a path by simply pressing a button?). To the best we can foresee, this technology will bring many new security perspectives.

I will not discuss on how the over-abundance of functionalities could affect security. Clearly, each functionality brings at least one more attack surface, each of us can imagine various scenarios.

In the long run, it is likely that the legal and ethical security concerns will be more critical than designing cryptographic security. However, one will be unable to tackle these more important issues as long as cryptographic security is not deployed on in-vehicle networks and components. So in the short run cryptographic security for automotive systems is an issue that needs to be urgently addressed.

Part III

References

Bibliography

[1] R. Anderson. On the security of digital tachographs. In *Computer Security-ESORICS 98*, pages 111–125. Springer, 1998.

[2] AUTOSAR. *Requirements on Crypto Service Manager, Part of AUTOSAR 4.2.1*, 2014.

[3] M. Bacchus, A. Coronado, and M. A. Gutierrez. The insights into car hacking. 2014.

[4] D. Basin, C. Cremers, and S. Meier. Provably repairing the iso/iec 9798 standard for entity authentication. *Journal of Computer Security*, 21(6):817–846, 2013.

[5] L. Bauer, L. F. Cranor, M. K. Reiter, and K. Vaniea. Lessons learned from the deployment of a smartphone-based access-control system. In *Proceedings of the 3rd Symposium on Usable Privacy and Security*, pages 64–75. ACM, 2007.

[6] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers. National Security Agency. THE SIMON AND SPECK FAMILIES OF LIGHTWEIGHT BLOCK CIPHERS. pages 16–45, 2013.

[7] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology-EUROCRYPT'94*, pages 92–111. Springer, 1995.

[8] S. Bittl. Attack potential and efficient security enhancement of automotive bus networks using short macs with rapid key change. In *Communication Technologies for Vehicles*, pages 113–125. Springer, 2014.

[9] M. Broy. Challenges in automotive software engineering. In *Proceedings of the 28th international conference on Software engineering*, pages 33–42. ACM, 2006.

[10] A. Bruni, M. Sojka, F. Nielson, and H. R. Nielson. Formal security analysis of the macan protocol. In *Integrated Formal Methods*, pages 241–255. Springer, 2014.

[11] O. Bubeck and V. Bourgeois. New security concepts for future generation automotive electronic control units.

[12] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *Proc. Royal Society of London. Series A, Mathematical and Physical Sciences*, 426(1871):233–271, 1989.

[13] C. Busold, A. Taha, C. Wachsmann, A. Dmitrienko, H. Seudié, M. Sobhani, and A.-R. Sadeghi. Smart keys for cyber-cars: secure smartphone-based nfc-enabled car immobilizer. In *Proceedings of the third ACM conference on Data and application security and privacy*, pages 233–242. ACM, 2013.

[14] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: A taxonomy and some efficient constructions. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 708–716. IEEE, 1999.

[15] L. Carettoni, C. Merloni, and S. Zanero. Studying bluetooth malware propagation. *IEEE Security & Privacy*, 2007.

[16] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, pages 197–213. IEEE, 2003.

[17] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security 2011*, 2011.

[18] A. Dardanelli, F. Maggi, M. Tanelli, S. Zanero, S. M. Savaresi, R. Kochanek, and T. Holz. A security layer for smartphone-to-vehicle communication over bluetooth. 2013.

[19] G. Delzanno and P. Ganty. Automatic verification of time sensitive cryptographic protocols. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 342–356, 2004.

[20] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in cryptology-Eurocrypt 2004*, pages 523–540. Springer, 2004.

[21] A. Fiat and M. Naor. Broadcast encryption. In *Advances in Cryptology (CryptoŠ93)*, pages 480–491. Springer, 1994.

[22] Freescale. *Freescale MC9S12XDP512 Data Sheet, Rev. 2.21*, 2009.

[23] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. Rsa-oaep is secure under the rsa assumption. In *Advances in Cryptology-CRYPTO 2001*, pages 260–274. Springer, 2001.

[24] I. Furgel and K. Lemke. A review of the digital tachograph system. In *Embedded Security in Cars*, pages 69–94. Springer, 2006.

[25] A. Greenberg. After jeep hack, chrysler recalls 1.4m vehicles for bug fix. *www.wired.com*, July 2015.

[26] A. Greenberg. Hackers remotely kill a jeep on the highway - with me in it. *www.wired.com*, July 2015.

[27] B. Groza and M. Minea. Bridging dolev-yao adversaries and control systems with time-sensitive channels. In *Critical Information Infrastructures Security*, pages 167–178. Springer, 2013.

[28] B. Groza and P.-S. Murvay. Broadcast authentication in a low speed controller area network. In *E-Business and Telecommunications*, pages 330–344. Springer, 2012.

[29] B. Groza and P.-S. Murvay. Efficient Protocols For Secure Broadcast In Controller Area Networks. *accepted for publication in: Industrial Informatics, IEEE Transactions on*, 2012.

[30] B. Groza, P.-S. Murvay, A. Van Herrewege, and I. Verbauwhede. LiBrA-CAN: a Lightweight Broadcast Authentication protocol for Controller Area Networks. In *Proceedings of The 11th International Conference on Cryptology and Network Security, CANS 2012, Springer-Verlag, LNCS*, 2012.

[31] B. Groza and S. Murvay. Secure broadcast with one-time signatures in controller area networks. In *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*, pages 371–376. IEEE, 2011.

[32] T. Guneysu, T. Kasper, M. Novotny, C. Paar, and A. Rupp. Cryptanalysis with copacobana. *Computers, IEEE Transactions on*, 57(11):1498–1513, 2008.

[33] C. T. Hager and S. F. Midkiff. Demonstrating vulnerabilities in bluetooth security. In *Global Telecommunications Conference, 2003. GLOBECOM'03. IEEE*, volume 3, pages 1420–1424. IEEE, 2003.

[34] H. Hartenstein and K. Laberteaux. *VANET Vehicular Applications and Inter-Networking Technologies*. Wiley, 2009.

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

[36] E. Haselsteiner and K. BreitfuSS. Security in near field communication (nfc) strengths and weaknesses. Technical report, Philips Semiconductors, 2006.

[37] D. Holcomb, W. Burleson, and K. Fu. Power-up sram state as an identifying fingerprint and source of true random numbers. *Computers, IEEE Transactions on*, 58(9):1198–1210, Sept 2009.

[38] C.-M. Huang, Y.-S. Chen, and I. Global. *Telematics communication technologies and vehicular networks: wireless architectures and applications*. Information Science Reference, 2010.

[39] Infineon. *Tricore TC1797 32-Bit Single-Chip Microcontroller, UserŠs Manual V1.1*, 2009.

[40] Infineon. *SP37T Datasheet.*, 1.0 edition, January 2010.

[41] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *19th USENIX Security Symposium, Washington DC*, pages 11–13, 2010.

[42] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *19th USENIX Security Symposium, Washington DC*, pages 11–13, 2010.

[43] T. Jeske. Floating car data from smartphones: What google and waze know about you and how hackers can control traffic. In *Proceedings of BlackHat Europe 2013*, MARCH 2013.

[44] F. Koeune and F.-X. Standaert. A tutorial on physical security and side-channel attacks. In *Foundations of Security Analysis and Design III*, pages 78–108. Springer, 2005.

[45] T. Kohno, A. Broido, and K. C. Claffy. Remote physical device fingerprinting. *IEEE Trans. Dependable Secur. Comput.*, 2(2):93–108, Apr. 2005.

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

[47] 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.

[48] K. Lemke, C. Paar, and M. Wolf. *Embedded Security in Cars Securing Current and Future Automotive IT Applications*. Springer Verlag, 2006.

[49] J. Leohold. Communication requirements for automotive systems. In *Keynote speech 5th IEEE international workshop on factory communication systems, Vienna, Austria, Vienna University of Technology*, 2004.

[50] C.-W. Lin, Q. Zhu, and A. Sangiovanni-Vincentelli. Security-aware modeling and efficient mapping for can-based real-time distributed automotive systems. 2014.

[51] G. Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 147–166. Springer, 1996.

[52] G. Lowe. Casper: A compiler for the analysis of security protocols. In *10^{th} Computer Security Foundations Workshop*, pages 18–30. IEEE, 1997.

[53] G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger. Nfc devices: Security and privacy. In *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, pages 642–647, March 2008.

[54] S. Malipatlolla and I. Stierand. Evaluating the impact of integrating a security module on the real-time properties of a system. In *Embedded Systems: Design, Analysis and Verification*, pages 343–352. Springer, 2013.

[55] M. Matsui and Y. Murakami. Minimalism of software implementation. In *Fast Software Encryption*, pages 393–409. Springer, 2014.

[56] C. Meadows. The nrl protocol analyzer: An overview. *The Journal of Logic Programming*, 26(2):113–131, 1996.

[57] S. B. Moon, P. Skelly, and D. Towsley. Estimation and removal of clock skew from network delay measurements. Technical report, Amherst, MA, USA, 1998.

[58] C. Mulliner. Vulnerability analysis and attacks on nfc-enabled mobile phones. In *Availability, Reliability and Security, 2009. ARES '09. International Conference on*, pages 695–700, March 2009.

[59] P.-S. Murvay and B. Groza. Performance evaluation of sha-2 standard vs. sha-3 finalists on two freescale platforms. *International Journal of Secure Software Engineering (IJSSE)*, 4(4):1–24, 2013.

[60] P.-S. Murvay and B. Groza. Source identification using signal characteristics in controller area networks. *Signal Processing Letters, IEEE*, 21(4):395–399, 2014.

[61] M. Naor and B. Pinkas. Threshold traitor tracing. In *Advances in Cryptology (CRYPTO'98)*, pages 502–517. Springer, 1998.

[62] T. Nolte, H. Hansson, and L. L. Bello. Wireless automotive communications. In J. Kaiser, editor, *Proceedings of the 4th International Workshop on Real-Time Networks (RTN05) in conjunction with the 17th Euromicro International Conference on Real-Time Systems (ECRTS05)*, pages 35–38. ISBN 3-929757-90-7, July 2005.

[63] A. Perrig. The biba one-time signature and broadcast authentication protocol. In *Proceedings of the Eighth ACM Conference on Computer and Communications Security (CCS-8)*, pages 28–37, Philadelphia PA, USA, 2001.

[64] A. Perrig, R. Canetti, D. Song, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Seventh Annual ACM International Conference on Mobile Computing and Networks (MobiCom 2001)*, pages 189–199, 2001.

[65] 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.

[66] R. Popescu-Zeletin, I. Radusch, and M. A. Rigani. *Vehicular-2-X Communication*. Springer, 2009.

[67] Portland Press Herald. U.S. military hackers scramble to fortify cars cyber defenses, November 2014.

[68] A. Radu and B. Groza. Security concept for smartphone car access via nfc rf id device. Technical report, Continental Software Conference, Regensburg, 2015.

[69] L. Reyzin and N. Reyzin. Better than biba: Short one-time signatures with fast signing and verifying. In *Proceedings of the 7th Australian Conference on Information Security and Privacy*, ACISP '02, pages 144–153, London, UK, 2002. Springer-Verlag.

[70] Robert BOSCH GmbH. *CAN Specification Version 2.0.*, 1991.

[71] Robert BOSCH GmbH. *CAN with Flexible Data-Rate Version 1.0*, 2012.

[72] T. Roeder, R. Pass, and F. Schneider. Multi-verifier signatures. *Journal of Cryptology*, 25(2):310–348, 2012.

[73] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*. National Institute of Standards and Technology, Revised: April 2010 Lawrence E Bassham III, Special Publication 800-22 Revision 1a.

[74] V. Shoup. Oaep reconsidered. In *Advances in Cryptology-CRYPTO 2001*, pages 239–259. Springer, 2001.

[75] V. Shoup. *A computational introduction to number theory and algebra*. Cambridge University Press, 2009.

[76] C. Solomon and B. Groza. Limon - lightweight authentication for tire pressure monitoring sensors. In *1st Workshop on the Security of Cyber-Physical Systems (affiliated to ESORICS 2015)*, 2015.

[77] F. Stajano. The resurrecting duckling. In *Security Protocols*, pages 183–194. Springer, 2000.

[78] D. Stebila, L. Kuppusamy, J. Rangasamy, C. Boyd, and J. G. Nieto. Stronger difficulty notions for client puzzles and denial-of-service-resistant protocols. In *Proceedings of the 11th international conference on Topics in cryptology: CT-RSA 2011*, CT-RSA'11, pages 284–301. Springer-Verlag, 2011.

[79] R. Steffen, J. PreiSSinger, T. Scholllermann, A. Mulller, and I. Schnabel. Near field communication (nfc) in an automotive environment. In *Near Field Communication (NFC), 2010 Second International Workshop on*, pages 15–20, April 2010.

[80] I. Studnia, V. Nicomette, E. Alata, Y. Deswarte, M. Kaâniche, and Y. Laarouchi. Survey on security threats and protection mechanisms in embedded automotive networks. In *Dependable Systems and Networks Workshop (DSN-W), 2013 43rd Annual IEEE/IFIP Conference on*, pages 1–12. IEEE, 2013.

[81] P. Syverson. A taxonomy of replay attacks. In *7th Computer Security Foundations Workshop*, pages 187–191. IEEE, 1994.

[82] C. Szilagyi and P. Koopman. Flexible multicast authentication for time-triggered embedded control network applications. In *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*, pages 165–174. IEEE, 2009.

[83] 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.

[84] C. J. Szilagyi. *LOW COST MULTICAST NETWORK AUTHENTICATION FOR EMBEDDED CONTROL SYSTEMS*. PhD thesis, Carnegie Mellon University, 2012.

[85] A. V. Thiemel, D.-E. M. Janke, and D.-E. B. Steurich. Speedometer manipulation-putting a stop to fraud. *ATZelektronik worldwide*, 8(2):16–19, 2013.

[86] S. Tillich and M. Wójcik. Security analysis of an open car immobilizer protocol stack. In *Trusted Systems*, pages 83–94. Springer, 2012.

[87] G. Tipa and B. Groza. High quality randomness for safety critical tasks on automotive embedded devices, research report. Technical report, 2014.

[88] G. Tipa, B. Groza, and R. Ragobete. Generation of random numbers from an uninitialised memory unit (European Patent Application EP 14465511.5-1953/28.05.14. on behalf of Continental Corporation). Technical report, 2014.

[89] A. Toth. Method and system for monitoring a parameter of a tire of a vehicle, Apr. 16 2014. EP Patent App. EP20,120,464,019.

[90] A. Toth. Method and system for monitoring a parameter of a tire of a vehicle, Apr. 16 2014. EP Patent App. EP20,120,464,019.

[91] A. Van Herrewege, D. Singelee, and I. Verbauwhede. CANAuth-a simple, backward compatible broadcast authentication protocol for can bus. In *ECRYPT Workshop on Lightweight Cryptography 2011*, 2011.

[92] 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.

[93] A. Van Herrewege, V. van der Leest, A. Schaller, S. Katzenbeisser, and I. Verbauwhede. Secure prng seeding on commercial off-the-shelf microcontrollers. In *Proceedings of the 3rd international workshop on Trustworthy embedded devices*, pages 55–64. ACM, 2013.

[94] P. van Zyl, S. v. Goethem, S. Jansen, S. Kanarchos, M. Rexeis, S. Hausberger, and R. Smokers. Study on tyre pressure monitoring systems (tpms) as a means to reduce light-commercial and heavy-duty vehicles fuel consumption and co2 emissions. *Final report, European Commission DG Clima*, 2013.

[95] P. Vasile, B. Groza, and S. Murvay. Performance analysis of broadcast authentication protocols on can-fd and flexray. In *WESS: 10th Workshop on Embedded Systems Security (affiliated to ESWEEK 2015)*, 2015.

[96] R. Verdult, F. D. Garcia, and J. Balasch. Gone in 360 seconds: Hijacking with hitag2. In *Proceedings of the 21st USENIX conference on Security symposium*, pages 37–37. USENIX Association, 2012.

[97] Q. Wang and S. Sawhney. Vecure: A practical security framework to protect the can bus of vehicles. In *Internet of Things (IOT), 2014 International Conference on the*, pages 13–18. IEEE, 2014.

[98] J. Wetzels. Broken keys to the kingdom: Security and privacy aspects of rfid-based car keys. *arXiv preprint arXiv:1405.7424*, 2014.

[99] M. Wolf. *Security engineering for vehicular IT systems*. Springer, 2009.

[100] M. Wolf, A. Weimerskirch, and C. Paar. Secure in-vehicle communication. *Embedded Security in Cars*, pages 95–109, 2006.

[101] S. Woo, H. J. Jo, and D. H. Lee. A practical wireless attack on the connected car and security protocol for in-vehicle can. 2014.

[102] M. Xu, W. Xu, J. Walker, and B. Moore. Lightweight secure communication protocols for in-vehicle sensor networks. In *Proceedings of the 2013 ACM workshop on Security, privacy & dependability for cyber vehicles*, pages 19–30. ACM, 2013.