Need to take cybersecurity into account in secure ECUs

Alin Mihalache, Fabrice Bedoucha

Institut pour la Maîtrise des Risques
Sûreté de Fonctionnement - Management - Cindyniques

Nécessité de prendre en compte la cyber sécurité dans les ECU sécuritaires
Need to take cybersecurity into account in secure ECUs

Alin Mihalache
Research and Development Division
PSA Groupe
Vélizy, France
alin.mihalache@mpsa.com

Fabrice Bedoucha
Research and Development Division
PSA Groupe
Vélizy, France
fabrice.bedoucha@mpsa.com

***Résumé*—Le but de cet article est de présenter les moyens mis en œuvre pour protéger les logiciels des calculateurs automobiles de potentielles attaques. Face aux risques cyber sécurité, quelles peuvent être les réponses des concepteurs ?**

***Abstract*—The goal of this article is to present the means put in place to protect automotive ECU from potential attacks. To manage cybersecurity risks, what can be the answers from designers?**

Keywords—cybersecurity, safety, automotive

I. INTRODUCTION

Today, car industry is not invulnerable to cybersecurity threats, many examples lately showed the vulnerability of the embedded systems to the cyber-attacks. There is always the risk that a hacker modifies the software and endangers the occupants of the vehicle. For example, to switch off light when driving, to block the steering column, to launch the airbags, etc.

The cybersecurity risks have all the more impacts when it is question of safety ECUs. Moreover, ISO 26262 standard starts to be interested in the cybersecurity.

The cybersecurity can concern several fields:

- Privacy – identification and tracking of vehicles or individuals;
- Financial – financial losses that may be experienced by individuals or ITS operators;
- Safety – impact on functional safety.

For example, it is possible that an attack has little or no impact on safety, but presents significant risks in terms of compromised driver privacy or loss of reputation for vehicle manufacturers.

In this article, we only take into account safety impacts.

After presenting several safety goals examples, we will describe the security standard in automotive, which is currently in the draft phase. Then, we will propose several examples of security gates. Finally, we will illustrate the cybersecurity process development and validation, before concluding.

II. SAFETY RISKS RELATED TO CYBERSECURITY

The embedded systems are more and more complex and safety risks are high especially with ECU that can react and replace the driver: decelerates when the vehicle comes closer to the front vehicle, switches on or switches off lights when luminosity is lower or higher than a threshold, activates wipers when it rains…

A. Safety goals ASIL QM

In this context, what can happen if the climate control system started blasting cold air at the maximum setting or the radio switched to the local hip-hop station at full volume or a family picture appear on the car's digital display?

The driver is certainly not happy, even if his life is not in danger.

B. Safety goals ASIL A or ASIL B

We continue our imagination. What can happen if the window windshield wipers suddenly turned on without being driven by the driver, and wiper fluid blurred the glass?

Or if the low beams switches off in the darkness on an unlit highway?

Surely, the driver begins to be in danger.

C. Safety goals ASIL C or ASIL D

What can happen if the accelerator stopped working or if the car accelerates without the driver's will? Or a not expected key off at a fairly high speed?

Or no control of the steering, brakes, and transmission…?

The driver is really in danger and risks his life.

All these events really happened when Chris Valasek and Charlie Miller [4] took control remotely of a car taking

advantage of several flaws in the conception; these dreaded events can be possible if the cybersecurity is not taken into account on the connected vehicle. A hacker can be able to send commands on the dashboard functions, steering, brakes, and transmission, all from a laptop that may be across the world. This wireless control, via the Internet, can control thousands of vehicles.

III. ISO/SAE 21434 CYBERSECURITY STANDARD

ISO and SAE have a good collaboration in the area of road vehicle. They are working for the first standard for automotive cybersecurity. This standard is ISO/SAE 21434 "Road vehicles – Cybersecurity engineering" [1], in the revision phase now, the final release is expected in 2020. At the present, substantial modifications to its content of the standard are still entirely possible. This standard should eventually replace SAE J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems. The SAE J3061 [2] specification was published in 2016 as a practical document with recommendations, providing an engineering process framework for integration with other development processes for the complete design of cybersecurity in on-board systems cars.

There are several benefits to have an automotive cybersecurity standard: defining a set of criteria for the cybersecurity engineering, common terminology, industry-wide consensus for the cybersecurity issues… in order to minimize contradiction between the different actors of the automotive world carmakers and suppliers.

A. Scope of the ISO/SAE 21434 standard

The ISO/SAE 21434 standard will be applicable to road vehicles: sub-systems, components, hardware and software. The main scope is to have a structured process in place in order to do a "security by design" process.

The life cycle chosen by ISO/SAE 21434 standard is the same as ISO 26262 [3], on the development process. During all the phases of the life cycle, the security aspects must to be considered. A secure vehicle is the consequence of the secure requirements implementation at the design phases.

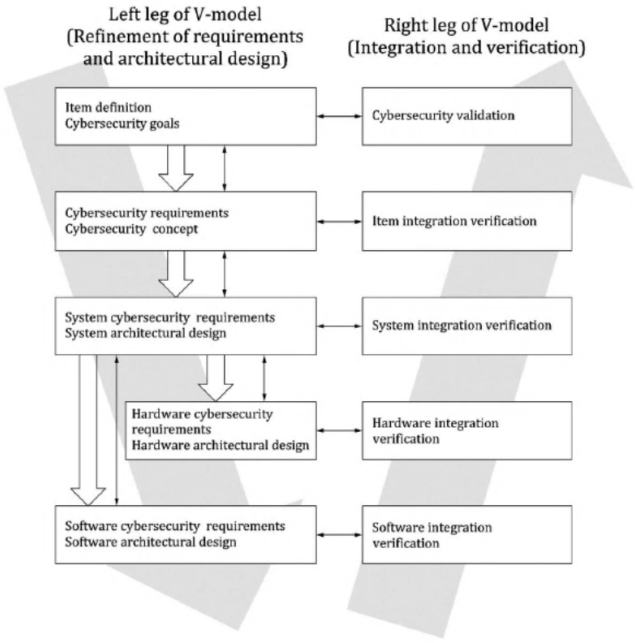

Fig. 1. Cybersecurity V cycle [1].

This standard will not explain the specific cybersecurity practices: the solutions or the technologies to be used. No recommendation like encryption methods, countermeasures…

B. Risk assessment

Establishing the reasonable high security risk level (and not the highest security risk level) is one of the main goal of the standard. The study of risk assessment start with a "threat analysis and risk assessment" (TARA). The equivalent of this process for safety is "hazard and risk analysis" (HARA).

Similar with HARA, the TARA is a methodology used for identification and assessment of the potential damage of the cybersecurity risk (attacks, threats and vulnerabilities). This methodology fix a set of the countermeasure in order to mitigate this risk. The risk level must to be estimated, issue from the damage scenarios, and must be reduced by the countermeasure (encryption for example) until the remaining risk level is acceptable.

In TARA methodology, several factors intervene to determine the risk. Not only the impact is considered but also if the vulnerability can be exploited by everyone or only by experts, if the vulnerability can be exploited remotely or a physical access to the car is needed, if the vulnerability concerns one specific car or all the cars of the same model…

A risk is lower when only experts can exploit a vulnerability, physical access is needed and it concerns only one specific car (for example, cryptographic keys are unique for each car). On the other hand, the risk is higher if everyone can exploit the vulnerability, remotely and it concerns all the fleet (example: cryptographic keys are the same for all cars).

C. Process phases and their relationship ISO 26262 and ISO/SAE 21434

The safety process (ISO 262626) is not sufficient to include the cybersecurity process particularity (ISO/SAE 21434). Each standard has its own process. The designers must by able to take into account both process.

Here are some similarities and differences between the two processes. To represent a system in a vehicle, ISO 26262 uses the term item while ISO/SAE 21434 uses the term feature. Therefore, ISO 26262 and ISO/SAE 21434 must be applied to the same ECU.

A cybersecurity attack to a critical system has the potential to produce a system failure. The effect produced is similar like a fault in a safety critical system. The concept of harm is the same in both standards, referring to physical injury or damage to the health of persons. The source of harm in safety as the hazard, while in cybersecurity it is the threat. HARA and TARA are phases in the lifecycle that are also similar, in the sense that they provide common techniques to mitigate a potential source of harm. HARA is the base for the definition of safety goals, and the provision of safety measures and TARA is used to define cybersecurity goals and provide cybersecurity measures.

In both standards, the goals are the top-level requirements, which are decomposed in more refined requirements during the lifecycle stages. When both standards are applied, all top-level requirement safety and cybersecurity are used to create the system architecture.

The both processes require a high-level system description that is used to generate preliminary architectural expectations. During system level, technical requirements are allocated into the system architecture which is then refined into a hardware and software architectural design. Process phases and their horizontal relationship between the both standards is presented in Fig. 2.

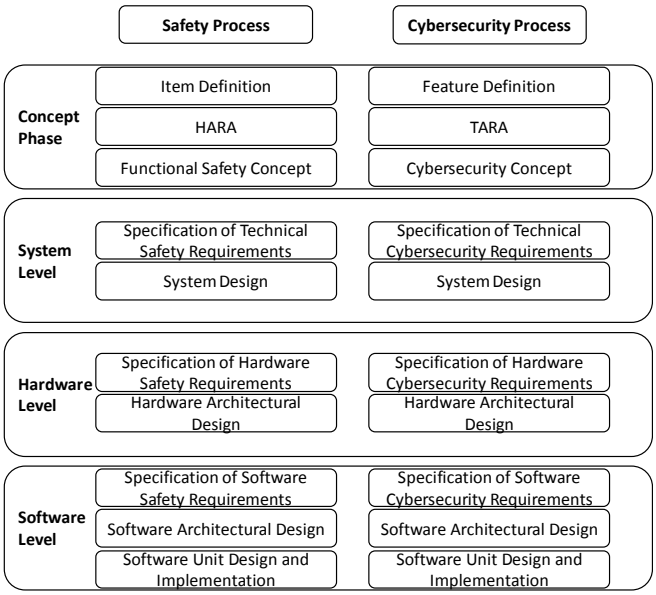

Fig. 2. Process horizontal relationship between the both standards

IV. CYBERSECURITY PROTECTIONS AT ECU LEVEL

In this part, we are going to describe the main cybersecurity protections that we can implement in a typical automotive ECU like Engine control or Body controller. We do not treat ECU linked with multimedia functionalities where protections must be hardened due to the different attack possibilities: Bluetooth, Wi-Fi…

In our case, we have an ECU with a microcontroller, which includes in the same chip memories for software and data. Software is executed from flash memory where it is stored (no copy in RAM) and network communication is limited to CAN (Controller Area Network) or LIN (Local Interconnect Network) which are typical in automotive.

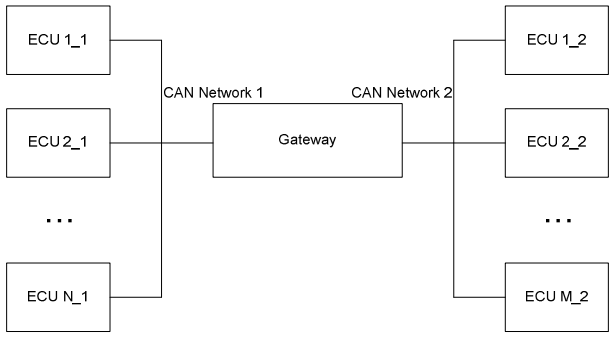

Fig. 3. Typical network architecture in automotive

A. *Key management*

In Cybersecurity, protection of assets is mainly based of cryptographic or signature keys. These keys are secrets to keep; otherwise, the security will be compromised.

It is possible to store keys in specific devices called HSM for Hardware Security Module. Typically, HSM is integrated into the microcontroller chip including the main core or cores for multicore microcontrollers. When a core needs to encrypt data for example, it sends the data to be encrypted to the HSM (through shared memory). Data is encrypted inside HSM with the cryptographic keys and encryption result is returned to the Core. In this way, cryptographic keys are kept safe into HSM, the Core does not know theirs values and does not have access to them. Moreover, HSM allows fast operation by integrating hardware accelerators i.e. AES 128 for encryption and decryption.

Once cryptographic and signature keys are stored in HSM, they are not accessible by attackers. However, we must consider two things. First, before the storing of keys in HSM, precautions must be taken to avoid the keys to be accessible. This is out of the scope of the article but storage of keys in computer and transfer to ECU must be also robust to attacks. A second point is the degree of protection of HSM. Unless there is a known vulnerability of the microcontroller, it would be very time consuming and expensive for attackers to retrieve keys in HSM and it may not be successful. However, if the keys are the same for all the ECU, it may be worth for attackers to try to read inside HSM because there will need to do the work once, the ratio cost versus benefits may be interesting for attackers. However if the keys are different, attackers will need to repeat a high cost operation without being sure that it will work. Therefore, although management of the keys will be more difficult, it is better to have different keys for each ECU. For example, to encrypt one secure data in an ECU, it is much more secure to use different cryptographic key in each ECU. In ECU1 in Vehicle1, encryption/decryption is done with Key1 and in the same ECU1 in Vehicle 2; encryption/decryption of the same data is done with Key2. Last point, the choice of the keys must be done randomly so that it cannot be possible to determine Key2 in vehicle2 knowing Key1 in vehicule1.

B. *Secure boot*

The aim is to ensure that only a known software is running in the ECU. Before starting the software, it is checked that it is a valid one and not a tampered one. One typical way to check a software is to write its CRC at the end of the software in flash memory and at each start-up, the CRC is recalculated and compared with the one written in flash memory. This way is not enough to assure cybersecurity. Indeed, calculating the CRC of the flash is more to check the integrity of the flash than checking the validity of the software. Moreover, CRC is easy to crack; an attacker is able to change the software and easily calculating its CRC.

To have a strong secure boot, software must be signed using a ciphered algorithm for example AES-CMAC-128. Like CRC, at each start-up, the signature of the software is calculated and compared with the one stored in HSM. Therefore, HSM must store at least the cryptographic key to calculate the signature and the signature of the software.

If the signature calculated is the same as the one stored in HSM, it means that the software is valid and can be launched. Otherwise, it means that the software is tampered and for safety reasons it must not be run. If necessary, a safe state could be launched. For example, if the airbag software is detected as tampered, airbag software is blocked, so airbag explosion is not possible and it is reported to the driver that airbag system is defective.

Timing performances depends on the microcontroller itself, its frequency, the size of flash memory and how hardware accelerator is designed. In one of our ECU, secure boot last around 250ms for a software of 7 Megabytes. As secure boot is launched at each start-up, the ECU is not ready as soon as it starts up. First, it is initialized then secure boot is launched. In our previous example, we have to add 50ms more, so the ECU is ready to work after 300ms. For some application, it is not a problem to wait 300ms before being operational but for other ECU, it can be critical. Therefore, when we have a secure boot, we can run in two different modes: serial mode and parallel mode. In serial mode, we have time and secure boot goes until its end before using the ECU. In parallel mode, we cannot wait for the end of secure boot due to strong timing constraints and software is starting before the end of secure boot, which runs in parallel in HSM. If after secure boot, the result is that the software is tampered, HSM blocks the software, which is running.

Serial mode is more secure than parallel mode because a possible tampered software is not executed at all whereas in parallel mode, it is executed during a short time and it may have time to take advantage of vulnerabilities of the microcontroller for example. Therefore, when there is no strong timing constraints to start up an ECU, it is better to stay in serial mode.

C. Secure update

In order to have a robust secure boot, we must be sure that the update of the software is done with a valid software. Indeed, if we have a robust secure boot, we must have also a robust secure update. Otherwise, the attacker would download a tampered software seen as a valid software.

We must use a mechanism, which checks the validity of the software. For example, the new software is signed and before installing it, its signature is checked. Typical signature used is based on asymmetric keys working in pair: the private key and the public key. The software is signed with the private key, not known except by the authority responsible to sign the software and the signature is sent with the software. The ECU where the software is installed calculates the signature with the public key, which by nature is known. The result must be the same. If the result are not the same, it means that it is not a valid software to install and the installation is aborted.

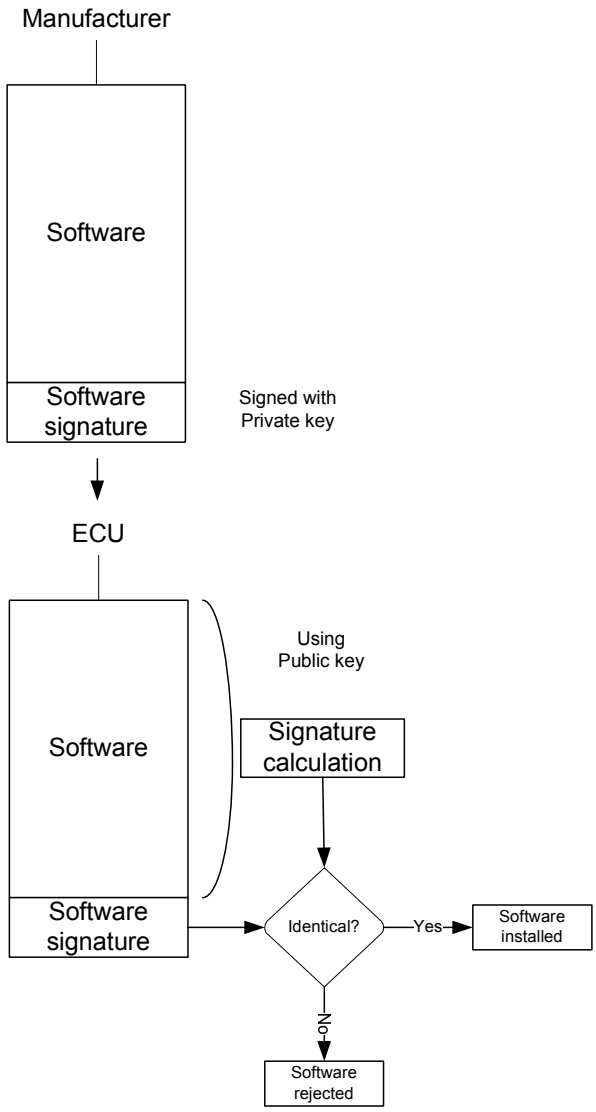

Fig. 4. Example of a signature verification

This description does not mention the use of certificates and PKI (Public Key Infrastructure) which must be used to improve security. Moreover, for ECUs that can be downloaded over the air, the transmission must be secured as well as back-end servers. This topic is out-of- the-scope of this paper.

The performances with the same ECU mentioned previously is around 1.5 second to check signature of 7 Megabytes. Calculation in the HSM with hardware accelerator. The timing is not critical in this use case because the ECU is not running; we are either in factory or in the garage to download a software in the ECU.

In addition, most of the time in automotive, we have the program memory inside the microcontroller; there is no need to encrypt the software to protect it. However, the binary file should be protected because if it is in the hands of attackers, they can disassemble it to understand how the software works and maybe find some vulnerabilities.

D. Secure storage

If we have safety data stored in a non-volatile memory, typically EEPROM, which is outside the microcontroller, these data must be protected. We could use a CRC but once

again, a CRC is not enough to assure cybersecurity, it can be easily cracked. The same as secure boot, the best solution is to sign them and if we also want to keep the safety data not understandable, they may be encrypted. For each cybersecurity functionality, we must use different keys and not always the same key.

E. Secure debug

For software team, it is often necessary to be able to debug its software even during production. This possibility must not be a backdoor for hackers.

For example, JTAG (Joint Test Action Group) implements standards for on-chip instrumentation. It specifies the use of a dedicated debug port implementing a serial communications interface, which is present in most of microcontrollers used in automotive. Thanks to this link, developers can have access to the whole memory, the registers; put some breakpoints and record parts of code executed. This is very useful for developers; however, this access is very dangerous in term of cybersecurity. It would even be possible to retrieve cryptographic keys through this interface. To avoid that, protections like authentication before accessing to debug function must be put in place or simply this debug possibility should not be possible when ECU is in mass production.

F. Communication

In automotive, CAN networks are mainly used. When a receiver receives a message, it does not know who the sender is. Indeed any node of the network can send a message. We can imagine that an ECU supposed to send a message is disconnected and instead a tool sends the message in its place. It allows the attacker sending wrong message that is considered correct for the receiver. We can also imagine that the software of an ECU is compromised and the attacker has reprogrammed it to send specific messages with wrong data and possible impacts on safety. In this case, the receiver is not able to detect that received data is wrong.

E2E (End-to-End) transmission is a known way to protect data in a frame by adding counter and checksum or CRC in the frames. However, the aim of E2E transmission is more to check that there is no transmission error instead of avoiding an attack. Attackers can reproduce E2E transmission easily.

To avoid attacks, we can use a more sophisticated system in CAN. For example, the sender must sign each message sent to the network. The receiver checks the signature before taking the data of the message into account. Of course, the signature for each sent frame must be different in order to avoid a replay attack where the attacker records an exchange between two ECUs and replay it. Therefore, a specific algorithm based on cryptographic keys must be put in place.

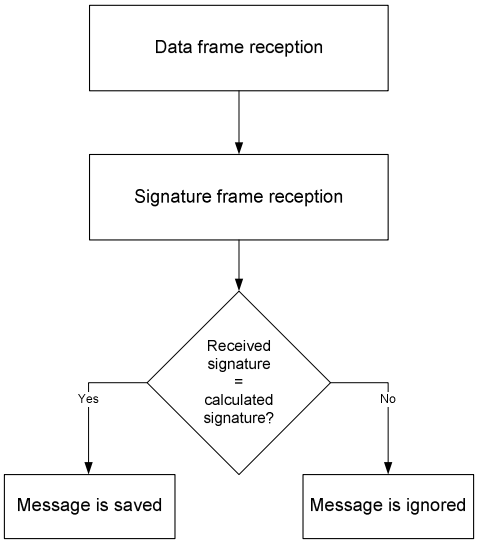

Fig. 5. Example of signed frame exchange.

Moreover, in automotive many current works emerge to propose IDS (Intrusion Detection System) on CAN. The aim is to detect messages that are not supposed to be received. For example, a message is supposed to be periodically sent every 100ms and it is received after 20ms of the previous one. Then it is reported and can be treated as a fake message.

Some ECUs can have other possibilities to communicate. For example, Ethernet is more and more used in automotive, especially with ADAS (Advanced driver-assistance systems such as emergency breaking, line-keeping assist…) systems and Multimedia. Ethernet has also its weaknesses that can be compensate by IDS and firewall.

Moreover, some ECUs can also have wireless connections like Bluetooth, WI-FI or 2G/3G/4G. These connections must be highly protected because these ECUs can be accessed remotely. Usually protections are similar to the ones we have in a computer.

V. ECU DEVELOPMENT AND VALIDATION

The more the cybersecurity level is high, the more it is expected actions on specification, development and validation. The CAL (Cybersecurity Assurance Level) can be used to determine with which level of rigor cybersecurity activities are performed. There are four levels from CAL1 to CAL 4, where CAL 4 is the highest level. There is no specific rule to determine CAL but its level could be determined according to the impact on the attack and the attack vector, for example physical where you need to have access to the ECU or remotely.

Cybersecurity process is very similar to safety process. Therefore, we will not go into details of development and validation activities, to focus on specificities.

When we talk about validation in cybersecurity, like other domains, it is especially based on validating requirements. However, in cybersecurity, flaws can be revealed during penetration test, or pentests. This is specific to cybersecurity where a cyberattack is simulated. The test is performed to identify both weaknesses and strengths of the system. The aim is to evaluate the risks of a cyberattack and its consequences. Knowing the results of pentests we can takes actions to improve the security of our system. In

automotive, pentests can be executed on a specific ECU, or a group of ECUs related to a specific functionality or on a car itself.

Pentests allow the knowledge of the vulnerabilities and their consequences. New requirements can emerge following a pentests campaign.

Main objectives is to know if the keys are not accessible, if it is not possible to download a non authorized software or to boot on a malicious software, that access to debug functionalities is not possible…

During pentests, one specific technic is to perform Fuzzing testing where random data are provided as inputs. Embedded software must be robust and avoid crashes or memory leaks. In communication, random messages can be sent automatically as input of a system, which must behave without exploitable security flaws.

VI. CONCLUSION

Cybersecurity will soon have its norm in automotive where methodology is similar to safety and ISO 26262. As well as safety, cybersecurity must be taken into account at the start of the development.

There are many possibilities to protect ECUs from malicious software. Not all of them must be used in each ECU; it depends on the cybersecurity analysis and risk analysis. These protections can be complex to implement and of course have a cost.

Cybersecurity must be taken into account to maintain the safety. Applying only safety methods is not sufficient because these methods will prevent from failures, not from the intrusion of hacker.

ACKNOWLEDGMENT

The authors acknowledge the contribution of their colleagues to this work.

GLOSSARY

ECU: Electronic Control Unit

ISO: International Organization for Standardization

SAE: SAE International, previously known as the Society of Automotive Engineers

CAL: Cybersecurity Assurance Level

ADAS: Advanced Driver-Assistance Systems

E2E: End to End

PKI: Public Key Infrastructure

IDS: Intrusion Detection System

CRC: Cyclic Redundancy Check

JTAG: Joint Test Action Group

AES-CMAC-128: Advanced Encryption Standard - Cipher-based Message Authentication Code – 128

HSM: Hardware Security Module

CAN: Controller Area Network

HARA: Hazard And Risk Analysis

TARA: Threat Analysis and Risk Assessment

REFERENCES

[1] ISO/SAE DIS 21434 Road vehicles — Cybersecurity engineering, draft, https://www.iso.org/standard/70918.html.

[2] SAE J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems, 2016, https://www.sae.org/standards/content/j3061_201601/.

[3] ISO 26262 Road vehicles — Functional safety, 2018, https://www.iso.org/standard/43464.html.

[4] C. Valasek and C. Miller, “Remote Exploitation of an Unaltered Passenger Vehicle”, August 10, 2015, http://illmatics.com/Remote%20Car%20Hacking.pdf.

[5] J.-P. Castellanos Ardila and B. Gallina, “Towards Efficiently Checking Compliance Against Automotive Security and Safety Standards”, October 23, 2017, https://www.researchgate.net/publication/319257123_Towards_Efficiently_Checking_Compliance_Against_Automotive_Security_and_Safety_Standards

[6] A. Mihalache, F. Bedoucha, Y-M. Foll, M. Foucault, "Cohabitation de modules logiciel critiques et non critiques en utilisant des partitions et de la supervision", Lambda Mu 21, Reims, France, 2018.

[7] A. Mihalache, F. Bedoucha, Y-M. Foll, M. Foucault, "Assurer la SdF d'un logiciel existant ayant de nouvelles fonctionnalités spécifiées en Model Based Design", Lambda Mu 20, Saint-Malo, France, 2016.

[8] A. Mihalache, F. Bedoucha, "Norme ISO 26262 : application sur le logiciel du Boitier de Servitude Intelligent (BSI) de PSA", Lambda Mu 19, Dijon, France, 2014.

[9] A. Mihalache, "Modélisation et évaluation de la fiabilité des systèmes mécatroniques: application sur système embarqué", PhD thesis, ISTIA, Université d’Angers, 2007.