embeddedworld2020
Exhibition&Conference
... it's a smarter world

Safety-Security Co-Analysis with STPA: A Case Study on Connected Cars

Lijun Shan, Claire Loiseaux
Internet of Trust
Paris, France
{Lijun.shan, Claire.loiseaux}@internetoftrust.com

Nadja Marko, Joaquim Castella Triginer
Virtual Vehicle
Graz, Austria
{Nadja.Marko, Joaquim.CastellaTriginer}@v2c2.at

***Abstract*— Since modern vehicles are increasingly digitized and networked, security attacks may lead to malfunctions of vehicles on a massive scale. For instance, a cyber-attack through over-the-air (OTA) update may install malicious software on electronic control units (ECUs) in numerous vehicles, possibly resulting in severe accidents. To reveal security threats and their potential impact on safety, security analysis needs to be conducted in the early stage of automotive engineering. In ECSEL Joint Undertaking project SECREDAS we investigate a new approach to automotive safety-security co-analysis. This paper reports a co-analysis method which extends Systems Theoretic Process Analysis (STPA). Our method is evaluated with a case study of an OTA update system, which shows that the STPA-based method enables a close integration of safety and security analysis, as the entire socio-technical system is considered as a whole in achieving security and safety.**

Keywords—safety; security; connected cars; over-the-air update

I. INTRODUCTION

A modern car is an electro-mechanical system in which the electronic control units (ECUs) control the electrical systems for managing the engine, powertrain, transmission, brake, body, suspension, etc. A ***connected car*** is a car which is equipped with communication technologies to enable data exchange with external systems. With on-board devices such as telematics which communicate with OEM server, map servers, web server and mobile phones, a connected car provides various digital services, including remote vehicle maintenance, driving assistance, autonomous driving and infotainment.

One of the safety-critical services of a connected car is Over-The-Air (OTA) updating, which updates the on-board systems' software or the ECUs' firmware by downloading packages from OEM servers via the internet, regardless of the car's geographic location. Compared to the conventional means of software updating at the maintenance stations, OTA update brings the benefits of saving money, instant patching of critical bugs, and adding new features throughout the vehicle's lifecycle [1]. However, OTA update is at the risk of being compromised, which may result in numerous cars being updated with malicious software. How to develop an OTA system while ensuring the cars' safety and security is a challenging task not yet completely solved.

Safety-security co-engineering is an emerging discipline for developing safety-critical cyber physical systems (CPSs). At the concept phase of the engineering process, a safety-security co-analysis serves to reveal the interactions between the safety and the security of a system, to derive safety and security requirements, and to avoid duplicate workload. This paper proposes a safety-security co-analysis method based on the safety analysis technique STPA [2], and demonstrates the method with a case study of an OTA system of connected cars. The contributions are two-folds. First, in the methodological aspect, STPA is extended to analyze the safety and security of a system through a unified process. The proposed method depicts the control structure of the system-under-study and identifies possible safety hazards and security incidents by examining the design flaws in the system's process model, which helps to reveal the security-safety interaction comprehensively. The method also facilitates to explore a broad design space by evaluating different design options with explicitly specified assumptions. Second, the case study produced a set of high-level safety and security requirements on the OTA system. Compared to our previous security analysis on the OTA system [3] with the SAHARA method [4], the analysis with STPA-based method focuses on improving the system's operation process.

II. RELATED WORK

Safety analyses at the early stage of engineering process aim to identify safety hazards and to derive safety requirements. Safety analysis techniques, e.g. FMEA and FTA, have been proposed in 1960s and widely applied on safety-critical systems including automotive systems. Based on reliability theory, these safety analysis techniques attribute system safety hazards to component failures. However, such methods are insufficient for analyzing modern CPS systems e.g. connected cars, in which safety hazards are more often caused by system design defects or security vulnerabilities than by component failures.

To overcome the shortcomings of the conventional safety analysis techniques, Nancy Leveson proposed Systems-Theoretic Accident Model and Processes (STAMP), a new accident causality model based on systems theory. STAMP provides the theoretical foundation for a safety hazard analysis technique called Systems Theoretic Process Analysis (STPA) [2], which aims to identify a broad array of accident scenarios to

eliminate or control safety hazards. STPA models a system with control loops which may have both social and technical components and can be represented at varying levels of abstraction. This allows the analysis to be conducted at early development phases, independent of the system's technical architecture. STPA has been applied in various safety-critical industrial case studies including automotive [5] [6].

Researchers have made efforts to extend STPA for security analysis. Nancy Leveson and William Young proposed STPA-Sec, an integrated approach to safety and security analysis based on STPA [7]. However, STPA-Sec has some limitations for identifying unsafe control actions due to security attacks [8]. Furthermore, the combination of STPA-Sec with other approaches like FMVEA provides a unified analytical process called Systems-Theoretic Likelihood and Severity Analysis (STLSA) [9]. STPA-SafeSec, which is another safety and security analysis method, overcomes the limitations of STPA by introducing security constraints to the analysis and by mapping the abstract control level of the system [10].

Security threat analysis techniques for Information-Communication Technology (ICT) systems have been adapted to the security analysis of automotive. SAHARA [4] is a security-aware safety hazard analysis method for automotive. It complements safety analysis with a beforehand step of security analysis. Firstly, security threats to a system are identified by applying STRIDE [10] and prioritized with regard to their respective impact on safety. Then, safety analysis is conducted, while the safety-critical security threats identified in the first step are taken as a special category of safety hazards. In our previous work [3], we applied SAHARA to analyze an OTA system, and successfully produced 136 security threats.

The security of automotive OTA update has been investigated in recent years. Security analyses which revealed security vulnerabilities in such systems have been reported in [11] [3]. Uptane is an implemented secure OTA update framework with resilience against critical security attacks, including repository compromises, freeze attacks to the vehicle's update mechanism, compromises at a supplier's site [12]. Uptane adds strategic features to the state-of-the-art software update framework in order to address automotive-specific vulnerabilities and limitations. The commonality of the existing researches is that they firstly assume a technical architecture of the OTA system, and then patch the system with security technologies. Little has been studied on how to derive diverse designs to satisfy the system's safety and security goals, and how to evaluate different design options.

III. USE CASE: OTA UPDATE SYSTEM

The system under study, as shown in FIGURE 1, consists of an OEM server and a connected car. Note that this model excludes the driver because the OTA process assumed in our case study requires little participation of the driver. The system is a hypothesized model inspired by the related work [1] [9].

Within the car, the ECUs are the update targets and are inter-connected via an In-Vehicle Network (IVN) e.g. CAN bus. A *Telematics* enables the telecommunication between the car and the external systems via the internet. A *Central Gateway* bridges the Telematics and the IVN. The Gateway also takes the role of OTA manager.

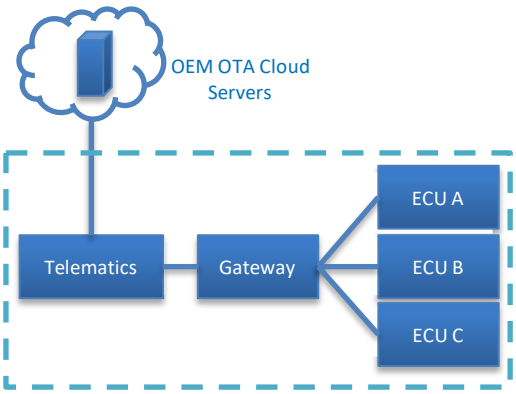

FIGURE 1 OVER-THE-AIR UPDATE SYSTEM

An OTA updating process involves two stages:

(1) Downloading: The OEM server pushes an update package to a car via the Internet. The package includes the update images i.e. the executable firmware of ECUs, and the metadata e.g. the version of the images. The Telematics receives the package and passes it to the Gateway, so that the Gateway can store the package in its local memory.
(2) Updating: When rebooting the ECUs, the Gateway firstly distributes the package to the target ECUs. Once the update has finished, the Gateway notifies the driver and OEM Server. Then the ECUs are rebooted with the new firmware.

The OTA system takes the "in place" approach [1], where only one version of the firmware exists on an ECU, and the target ECUs cannot be updated while they are in normal operation. To limit the scope of the study, we make the following assumptions:

- **A-1**: Only ECUs are the update targets. The update of other components, e.g. Telematics, Gateway, infotainment systems, take different processes and are beyond the scope of the analysis.
- **A-2**: The update packages are bug-free when released from the OEM. Malfunction of ECUs after being updated with an authenticate but problematic package is out of the scope of the analysis.

IV. OVERVIEW OF THE APPROACH

This section summaries the STPA technique through a step-wise process, and presents our extensions to STPA for the safety-security co-analysis.

A. Identifying safety accidents and hazards

STPA defines a safety ***accident*** as "an undesired or unplanned event that results in a loss, including loss of human life or human injury, property damage, environmental pollution, mission loss", and a ***hazard*** as "a system state or set of conditions that, together with a particular set of worst-case environment conditions, will lead to an accident (loss)" [2]. The identified system accidents and system hazards enable to trace lower-level analysis results to their system-level impact. For a connected car, an example of safety accident is "*the vehicle*

collides with objects or people when it is unable to brake". This accident may be caused by various safety hazards, e.g. a mechanical component of the braking system malfunctions, or an ECU in the braking system fails to function after being updated with a tampered image.

Extension. Our method identifies *security incidents* in parallel with safety accidents and hazards. In connected cars, security incidents are possible causes of safety accidents. Such cause-effect relations are also identified in this step. Our method of identifying security incidents is inspired by the security analysis methodology EBIOS [14]. A ***security incident***, called *feared events* in EBIOS, is the breach of a *primary asset*'s security criterion. A primary asset is either a ***function*** of a system or its hardware/software components, or some ***data*** stored, processed or transmitted in the system. The usual information security criteria include *availability, integrity,* and *confidentiality*. These criteria are exemplary rather than exclusive. We leave the other security criteria, e.g. Authenticity, Non-reputability, Authorization, for future research. A security incident may cause safety accidents or not, depending on the impaired assets' functionality. For example, the malfunction of an ECU in the car's braking system is a security incident because the integrity of the ECU's function is breached, and also a safety hazard which may lead to a safety accident. In contrast, an unauthorized modification to the mileage data stored in a car is a security incident but not a safety hazard. Our extension to STPA in this step allows to identify security incidents as one possible cause of safety accidents.

B. Defining actors and control actions

With STPA, a system is modelled by its functional control structure comprising hierarchical control loops. Actors in the control loops fall into two categories: ***controllers*** which execute the operation logic and take decisions upon specific conditions, and ***controlled processes*** which execute their controllers' commands and send feedbacks to controllers. It is worth noting that a controlled process can be refined into a control loop which further comprises a set of controllers and controlled processes. For example, at the top level, an OTA system can be modeled with a control loop consisting of two actors: an OEM server and a connected car. The system functions when the actors take ***control actions***: the server *pushes* update packages to the car, and the car *notifies* the server of the update's finish.

Extension. Our method extends control actions with parameters which represent the data transmitted by the actions. In the original STPA, control actions indicate real-time control signals, e.g. a driver *brakes* a car. Such an action is either executed as expected or not, depending on the action provider. In an ICT system, however, control actions are often data transmissions via communication channels, e.g. the internet and IVNs. The expected execution of a control action relies not only on the action provider but also on the communication channel. The distinction between a control action and the transmitted data helps distinguish different cases of the unexpected behavior of a control action, which can be an improper provision of the action or a compromise of the data.

C. Identifying hazardous control actions

With STPA, erroneous interactions and flawed safety constraints are regarded as the major causes of accidental scenarios. These defects are captured by identifying ***hazardous control actions***, i.e. improperly conducted control actions which may result in safety hazards. STPA suggests a set of heuristic keywords for deriving hazardous control actions from control actions: *Not provided*, *Provided* (*when not supposed*) *Provided too late/early*, *Provided out of order, Stopped too soon*, or *Applied too long*. Each identified hazardous control action is then associated to its consequences which are among the safety hazards identified in Step A.

Extension (1). Since a control action in ICT systems is data transmission, the data can be compromised during the transmission even if the actor has well conducted the action. We add two new keywords for deriving hazardous control actions *Provided but not received*, denoting the case that the control action is performed by the sender but the transmitted data is not received by the intended receiver; *Provided but impaired*, denoting the case that the control action is performed but the received data is compromised.

Extension (2). For safety-security co-analysis, we need to reveal the hazardous control actions' impacts not only on the system's safety but also on the security. In this step, our method associates the identified hazardous control actions to their consequential security incidents. For example, after the OEM server has pushed an update package, the package may be compromised during the transmission. This hazardous control action can be identified by applying the keyword *Provided but impaired* on the control action *push*(*package*). Consequently, the ECUs may be updated with compromised or camouflaged images, which is both a security incident and a safety hazard.

D. Identifying scenarios

The last step of STPA investigates the causes of hazardous control actions within specific scenarios, which are often flaws in the process model due to lacking of feedbacks. Two types of scenarios must be considered: (1) Why would a hazardous control actions occur? (2) Why would a control action be improperly executed or not executed? Then a set of safety requirements are derived as mitigations to such design flaws.

V. CASE STUDY

This section summarizes our case study on the OTA system with the extended STPA method.

A. Safety accidents and security incidents

*1) **Safety accidents and hazards***

TABLE 1 SAFETY ACCIDENTS OF CONNECTED CARS

	Safety Accident
A-1	Vehicle collides with objects or people, for instance when it is not able to brake, or when autonomous driving controller malfunctions.
A-2	Vehicle occupants are injured without vehicle collision, for instance when battery is too hot or even burns, or the doors cannot be opened or cannot be closed.

In this case study we consider only the accidents and hazards caused by electrical/electronic (E/E) systems' malfunctions, which may be a consequence of unsafe updating of ECUs. Other causes of safety accidents, e.g. failure of mechanical components, are beyond the scope of the analysis. TABLE 1 summarizes the identified safety accidents of connected cars, including a vehicle's collision and non-collision accidents.

In connected cars, the malfunction or unavailability of a safety critical E/E system, e.g. the braking system or the autonomous driving controller, may cause safety accidents. Tampered data, e.g. the impaired map or GPS data in an autonomous car, may also cause accidents. Even the unpredicted behavior of a non-safety-critical system e.g. an air conditioner or an infotainment system may distract the driver and hence cause safety accidents. The malfunctions of the E/E systems, either abnormal behavior or unavailability of functions, may be caused by unsafe update on their ECUs.

We identified three hazardous cases, concerning the update process, the package, and the occurrence of updating, respectively. TABLE 2 shows the safety hazards related to OTA update as well as their possible consequences in terms of safety accidents.

TABLE 2 SAFETY HAZARDS OF CONNECTED CARS

	Safety Hazard	Safety Accident
H-1	ECUs function abnormally or fail to function due to an unsafe updating ***process***, e.g. incorrectly designed process or incomplete execution of the updating process	A-1, A-2
H-2	ECUs function abnormally or fail to function after updated with camouflaged or tampered ***package***	A-1, A-2
H-3	ECUs function abnormally or fail to function due to a ***missing*** update, after a safety or security vulnerability being revealed	A-1, A-2

2) *Security incidents*

In a connected car, the assets include all the data processed or stored in the car including user credentials, system log, map data and mileage, as well as all the functions of the systems/components, including ECUs, telematics, gateway, IVNs and infotainment systems. In general, the security incidents of a connected car include any security breach which may happen to any of these assets. In this case study, however, we are only concerned with the security incidents as ***effects*** of unsafe or unsecure OTA updates. In the OTA update system, the major assets which may be impacted by a compromised update are the ECUs' functions.

TABLE 3 SECURITY INCIDENTS OF CONNECTED CARS

	Security incident	Safety accidents
I-1	(Non-availability of function) ECUs stop to function	A-1, A-2
I-2	(Non-integrity of function) ECUs function abnormally	A-1, A-2
I-3	(Non-confidentiality of function) ECU firmware is exposed without authorization	Non

Security incidents are derived by associating the violation of the security criteria *integrity, availability* and *confidentiality* to the asset. TABLE 3 summarizes the identified security incidents, and their possible impacts on safety are shown in the column "safety accidents". In TABLE 3, I-1 and I-2 are security incidents and safety hazards, while I-3 is only a security incident. From the security point of view, a security incident should be identified and mitigated, regardless of its impact on safety.

B. *Actors and control actions*

A control diagram is constructed to model the OTA system, as shown in FIGURE 2. The control actions are numbered to denote their order in the two stages of an OTA process. At the top abstraction level, the OEM Server is the controller, and the Car is the controlled process. Within the Car, the Gateway is the controller, while the Telematics and the ECUs are the controlled processes.

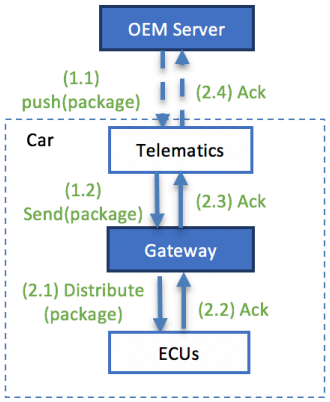

FIGURE 2 CONTROL DIAGRAM OF THE OTA SYSTEM

TABLE 4 summarizes the actors and their controlled actions.

TABLE 4 ACTORS AND CONTROL ACTIONS

Actors	Control Action	
OEM server	Push(package)	Sends an update package to a car
Tele-matics	Send(package)	Transmits the update package to the Gateway
	Ack(updated)	Transmits ACK to OEM server on receiving "update finished" from Gateway
Gateway	Distribute(images)	Distributes images to the target ECUs
	Ack(updated)	Acknowledges "update finished" message to Telematics on receiving ACK from ECUs
ECUs	Ack(updated)	Acknowledges "update finished" message to VCU

C. *Hazardous control actions*

This step identifies hazardous control actions and their possible consequences as safety hazards or security incidents. Hazardous control actions are derived by applying a set of keywords to each of the control action. Then each hazardous control action is associated to the safety hazards or security incidents identified in the previous step.

TABLE 5 summarizes a part of our analysis on Gateway. Take *Distribute(package),* one of the control actions of *Gateway,* as an example. One of the hazardous control actions derived is *Distribute(package) - Not Provided*, which results in a missing update i.e. the safety hazard *H-3* in TABLE 2. Note that the top row of TABLE 5 contains only the subset of the keywords of which the corresponding hazardous control actions raise safety hazards or security incidents. Those hazardous control actions which have impact over neither safety nor security are omitted. TABLE 5 shows that a hazardous control action may cause both a security incident and a safety hazard, or only one of them, or neither of them. For example, as the possible effect of a hazardous control action *Distribute(package) – Provided when not supposed*, a consequential safety hazard is *H-2 ECUs function abnormally or fail to function after updated with camouflaged package*. Meanwhile, this safety hazard is also a security incident because the ECUs as information assets have their security breached.

TABLE 5 HAZARDOUS CONTROL ACTIONS OF **GATEWAY**

Control action	Not provided / Not received	Provided when not supposed	Provided but impaired
Distribute (package)	H-3 Lack of updating	H-2 Updated with **camouflaged** package	H-2 Updated with **tampered** package
	I-1 ECUs' functionality unavailable	I-1 ECUs functionality unavailable	I-1 ECUs' functionality unavailable
		I-2 ECUs function abnormally	I-2 ECUs function abnormally
Ack(updated)	None	H-3 Lack of updating	None

D. *Identify scenarios*

In this step, we identify the scenarios where *process model flaws* cause hazardous control actions. Against the process model flaws, safety and security constraints are identified and transformed into requirements. The safety and security requirements can be evaluated in the control structure with respect to different scenarios and then be refined.

TABLE 6: SCENARIO ANALYSIS ON **GATEWAY- *DISTRIBUTE (PACKAGE)***

Hazardous control actions	Safety hazards or security incidents	Process model flaws	Safety (security) requirements
Not provided / Not received	*H-3 Lack of updating*	OEM incorrectly believes the car has properly updated the target ECUs	***Gateway-R-1*** Gateway shall check the versions of the updated ECU firmware and sends the info to OEM Server, either regularly (independent of updating) or immediately after an update
Provided when not supposed	*H-2 Updated with fake package*	ECUs incorrectly believe the images sent by Gateway are authentic	***Gateway-R-2*** Each ECU shall verify the authorization and version of its update image before updating
	I-1 ECUs function unavailable		
	I-2 ECUs function abnormally		
Provided but impaired	*H-2 Updated with compromised package*	ECUs incorrectly believe the images sent by Gateway are integrated	***Gateway-R-3*** Each ECU shall verify the integrity of its update image before updating
	I-1 ECU function unavailable		
	I-2 ECUs function abnormally		

TABLE 6 summarizes the results of this step concerning Gateway's action *Distribute*(*Package*). The analysis of the other control actions is omitted here for the sake of space. Among the safety requirements in TABLE 6, some are also security requirements. For example, ***Gateway-R-2*** and ***Gateway-R-3*** are also mitigations to security incidents ***I-1*** and ***I-2***. Since STPA analysis focuses on the process model, the produced safety and security requirements include to supplement the control structure with feedbacks. Different choices of adding feedbacks lead to a variety of design options, which helps to explore a wide design space.

For example, ***Gateway-R-1*** raises two design options: Gateway checks the versions of the updated ECU firmware and sends the information to OEM Server, either regularly i.e. independent of updating or immediately after an update.

If the latter solution is chosen, i.e. Gateway checks the firmware version of the updated ECUs after updating, again two possible mechanisms are possible:

- Stateless communication: The Gateway sends the ECU versions to the OEM server. Meanwhile the car launches before receiving the OEM server's response. This design brings better user experience with shorter vehicle downtime, but less safety.
- Stateful communication: Target ECUs are kept disabled before receiving the OEM server's confirmation. This design brings better safety with longer vehicle downtime.

Gateway-R-2 and ***Gateway-R-3*** raise the following two requirements:

- An update package contains a metadata which provides the information for the authorization/authentication and for integrity checking. This has been implemented in [10].
- ECUs verify the authority, authenticity and integrity of images before updating. This requirement is not always feasible because not all ECUs have the computing capability for this verification.

Gateway-R-2 and ***Gateway-R-3*** are both security requirements and safety requirements, as they address the hazardous control actions which may cause both security incidents and safety accidents.

VI. DISCUSSIONS

In the previous work [3], we have conducted a security analysis of the OTA system by applying SAHARA/STRIDE. The analysis started by depicting the system architecture in Data Flow Diagram (DFD), and produced 136 security threats at component level. TABLE 7 shows one of the identified threats, where *SecLev* indicates SAHARA Security Level.

TABLE 7 A THREAT TO CAN BUS

Threat Scenario	SecLev
Denial of service, for example this may be triggered on the internal network by flooding a CAN bus, or by provoking faults on an ECU via a malicious payload	3

Comparing our previous work with the analysis reported in this paper, the commonality is that both methods consider security incidents as one source of safety accidents. The differences lie in the following aspects:

Analysis process. In the previous case study with SAHARA, we conducted only the security analysis and left the safety analysis to safety experts. As the security and safety analysis are conducted in two separate stages which are black boxes to each other, it is unclear how well the security analysis results are used in the safety analysis and vice versa. In contrast, the STPA-based method allows simultaneous safety and security analysis, where the interactions between safety and security are considered throughout the analysis process. Starting with a high-level design, the STPA-based co-analysis requires little expertise on safety technologies and security technologies, which makes co-analysis a feasible task.

System modelling. SAHARA/STRIDE analysis is conducted based on an assumed system architecture. The analysis is hence limited to the technical design represented by the DFD model, aiming to improve the design by patching the components with security technologies. With STPA, the system is captured with its control structure which reflects the operation

logic rather than technical implementation. The system's process model is subject to be modified, which allows to explore a wider design space.

Analysis result. SAHARA/STRIDE analysis identifies technical threats to specific assets, e.g. a DoS attack to the CAN bus. With STPA-based method, the identified design defects are often caused by implicit assumptions of the design, e.g. ECUs incorrectly believe the images are authentic. Consequently, the security and safety requirements are concerned with the control logic, e.g. to add a feedback from a controlled process to its controller.

Overall speaking, the two methods are appropriate to two different stages of system engineering: STPA supports to derive high-level safety and security requirements at the first stage in order to explore the design space. SAHARA/STRIDE can be applied on a chosen system design to identify security threats to specific components.

VII. Conclusion

This paper reports a safety security co-analysis method based on STPA, and a case study of the OTA system of connected cars. Compared to the existing researches on secure OTA which aim to mitigate security threats with state-of-the-art technologies, our work focuses on identifying safety and security requirements in the early stage of system development. The hierarchical control structure provides a reference model to study the interactions between the components, and helps to conduct a comprehensive evaluation of different design options before choosing safety and security technologies. Our extension to STPA is light and naturally devised during the case study, showing that STPA is powerful and flexible to be applied to security analysis. The future work includes to improve the co-analysis method through case studies, and to further extend this method for safety security privacy co-analysis.

Acknowledgment

This work is supported by SECREDAS project (grant no. 783119) funded by the H2020-ECSEL program of the European Commission.

References

[1] Mckenna, Daniel; Automotive, BU; Semiconductors, NXP, "Making full vehicle OTA updates a reality," NXP White Paper, 2016.

[2] L. Nancy, Engineering a safer world: Applying systems thinking to safety, MIT Press, 2012.

[3] V. Alexandr, F. Stahl, H. Hamazaryan, Z. Ma, L. Shan, J. Kemmerich and C. Loiseaux, " "Practical Security and Privacy Threat Analysis in the Automotive Domain: Long Term Support Scenario for Over-the-Air Updates," in *VEHITS 2019 - Proceedings of the 5th International International Conference on Vehicle Technology and Intelligent Transport Systems*, 2019.

[4] M. Georg, E. Armengaud, E. Brenner and C. K. ".", "Threat and risk assessment methodologies in the automotive domain," in *Procedia computer science 83 (2016): 1288-1294.*, 2016.

[5] A. Abdulkhaleq, S. Wagner, D. Lammering, H. Boehmert and P. Blueher, "Using stpa in compliance with iso 26262 for developing a safe architecture for fully automated vehicles," arXiv preprint arXiv:1703.03657, 2017.

[6] S. Placke, J. Thomas and D. Suo, "Integration of Multiple Active Safety Systems using STPA," SAE Technical Paper No. 2015-01-0277, 2015.

[7] W. Young and N. G. Leveson, "An Integrated approach to safety and security based on systems Theory," *CoMMuniCations of the aCM,* 2014.

[8] C. Schmittner, M. Zhendong and P. Puschner, "Limitation and Improvement of STPA-Sec for Safety and Security Co-analysis," *Computer Safety, Reliability, and Security,* 2016.

[9] I. Friedberg, K. McLaughlin, P. Smith, D. Laverty and S. Sezer, "STPA-SafeSec: Safety and security analysis for cyber-physical systems," *Journal of Information Security and Applications,* no. 34, p. 183–196, 2017.

[10] A. Shostack, Threat modeling: Designing for security, JohnWiley&Sons, 2014.

[11] C. Thomas, E. Lesiuta, K. Rikley, C.-W. Lin, E. Kang, B. Kim, S. Shiraishi, M. Lawford and A. Wassyng, "Safe and secure automotive over-the-air updates," in *International Conference on Computer Safety, Reliability, and Security*, 2018.

[12] K. Trishank, A. Brown, S. Awwad, D. McCoy, R. Bielawski, C. Mott, S. Lauzon, A. Weimerskirch and J. Cappos, "Uptane: Securing software updates for automobiles," in *International Conference on Embedded Security in Car*, 2016.

[13] N. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety, The MIT Press, 2011.