Comparison of Safety and Security analysis techniques

Emilia Cioroaica[1], Smruti Ranjan Kar[2], and Ioannis Sorokos[1]

[1] Fraunhofer IESE, Kaiserslautern, Germany
{emilia.cioroaica,ioannis.sorokos}@iese.fraunhofer.de
[2] Magna Electronics Europe GmbH & Co. GmbH
Kurfuerst-Eppstein-Ring 9, Sailauf, Germany
SmrutiRanjan.Kar@magna.com

Abstract. Growing adoption of new technological advancements within the automotive domain is highlighting multiple safety and security concerns. Aiming at reducing the errors humans make on the roads, deployment of novel and intelligent technological solutions are likely to introduce multiple vulnerabilities that can be exploited by attackers. In automated driving scenarios, in particular, vehicles communicating with each other using ad-hoc networks are becoming vulnerable to specialized cyber-physical attacks. A single compromised vehicle will provide an attack entry point for all linked vehicles, putting lives of passengers at risk. In this paper, we present an overview and evaluation of safety and security analysis methods applied on a use case from the automotive domain.

Keywords: Safety, Security, Automotive, Analysis, Techniques

1 Introduction

Vehicles continue to transition into real cyber-physical systems. We can already connect vehicles to the internet and access their data with smartphones. In the future, vehicles will likely have even more options to connect: neighbouring vehicles, roadside infrastructure, and even smart homes could be potentially supported. In this way, automotive systems become part of a larger ecosystem which aggregates not only other systems, but human actors as well. Actors within automotive digital ecosystems, such as human drivers, OEMs (Original Equipment Manufacturers) and Tier-1/Tier-n companies are becoming increasingly concerned with the security vulnerabilities likely to threaten the safe operation of vehicles, resulting in general distrust within an ecosystem. Distrust in technology suppliers can affect the health of an ecosystem and negatively impact the business within. The ecosystem health is a measure of how well the business within an ecosystem is performing [19].

For addressing the emerging transition in the automotive domain, AUTOSAR [1] made an evaluation which showed that emerging requirements cannot

be realized by typical software architectures. Instead, they proposed an architecture for vehicles that is more flexible, highly available and capable to adapt over time to specific application requirements. In this way, the POSIX-based AUTOSAR platform enables a segregated delivery and execution of control functions on ECUs (Embedded Control Units) in order to facilitate the rapid development of safe autonomous driving. The flexibility of this emerging vehicle infrastructure supports the business growth within automotive smart ecosystems, by enabling vehicles to communicate and collaborate dynamically during operation. At the same time, this evolution brings additional safety and security concerns due to the communication channels being introduced by the comparably more open environment.

In this paper we are summarizing evaluation results of the state-of-the-art on safety and security analysis methods within automotive domains, focusing on collaborative vehicles. Modern vehicles are complex systems, considered to be a network of ECUs interacting with each other, and expected to accommodate runtime updates or adaptation of their functionality. ECUs within a vehicle may be connected to multiple internal networks where they perform different functions. For instance, some ECUs may serve as gateways that transform and share information between different types of networks. Architecture flaws can expose intra-vehicle information to inter-vehicle communication without an information barrier, resulting in vulnerabilities. Therefore, from a security attack perspective, such ECUs represent an effective target. Moreover, the safety-critical nature of a vehicle enables us to reason of the safety implications of malicious attacks. Last but not least, even though addressing safety considerations in the automotive domain is well-established, security considerations have also grown more prominent over the last decade.

In what follows, Section 2 provides a background on relevant regulatory approaches for developing automotive systems, Section 3 presents an overview of safety and security analysis from the current state-of-the-art, section 4 provides a summary of comparison between the methods and Section 5 presents conclusions and future work.

2 Background

Following the 2011 ISO 26262 recommendation of "reasonably foreseeable misuse" as a factor risk analysis [12], further elaborations of misuse with malicious attacks have emerged, with recommendations for security practices being published 9 years later [5]. Due to the fact that (a) at the moment when security practices started to emerge, safety was well established in the automotive domain and (b) security incidents can compromise safety, first security approaches have proposed a joint addressing of safety and security at all stages of system development, initially in a component-oriented fashion [18]. As reported in [17], **safety** is concerned with *preventing accidents* through identification of potential weaknesses, events, internal hazards and potentially hazardous states, followed by identification and implementation of appropriate mitigation mechanisms to reduce the risk to a

tolerable level. On the other hand, **security** is concerned with *protecting assets* against internal or external threats and vulnerabilities that can compromise them. A mechanism for protecting assets is through the use of control strategies that reduce the risk of compromising the functionality to an acceptable level.

Driven by the first approach to use a component-level perspective for combined hazard and threat analysis as a safety & security mechanism performed during concept design, popular safety analysis methods were further extended to include security analysis as well. For example, FMEA (Failure Modes and Effects Analysis) [14] was extended towards FMVEA (Failure Mode, Vulnerabilities and Effects Analysis) [16], HARA (Hazard Analysis and Risk Assessment) [20] was extended towards SAHARA (Security-Aware Hazard and Risk Analysis) [9], STPA (System Theoretic process Analysis) [4] towards STPA-Sec (Systems-Theoretic Process Analysis for Security) [21].

3 State-of-the-Art

In this section we provide an overview of the state-of-the-art safety and security analysis methods, presenting them in a chronological order. Our work provides a complementary analysis to the threat analysis and risk assessment methods presented in [7] by looking at the characteristics of methods that perform a combines safety & security analysis from different viewpoints as are presented in Section 4.

3.1 CHASSIS

Combined Harm Assessment of Safety and Security for Information Systems (CHASSIS) [15] was introduced in 2012 for supporting the derivation of safety and security requirements of a system. For an already developed system, design artefacts can be reused for CHASSIS as well, otherwise, if the system is in the design phase, safety and security analysis can be performed in parallel. During CHASSIS, functional requirements artefacts, such as use cases and sequence diagrams, are analyzed with HAZOP (Hazard and Operability studies) keywords. The analysis results in a set of safety and security requirements documented in UML (Unified Modelling Language). During elicitation of functional requirements, D-UCs (Diagramatic Use Cases) and T-UCs (Textual Use Cases) are used for describing users, functions and services, whereas UML sequence diagrams are used to model sequences of interactions between objects. During the elicitation of safety and security requirements, potential misuses of the system are identified in brainstorming sessions that involve domain experts as well as safety and security experts. Misuses are identified through a combination of use cases and HAZOP guide words, documented in D-MUC (Diagrammatic Misuse Cases), T-MUC (Textual Misuse Cases) and MUSD (Misuse sequence Diagrams). Failures are documented in FSD (Failure-Sequence Diagrams).

3.2 FMEVA

Failure Mode, Vulnerability and Effect Analysis Method [16] was introduced in 2014 as a security-oriented extension of the FMEA (Failure Mode and Effect Analysis) with the scope of unifying safety and security cause-effect analysis. Extending the analysis of a given *failure mode*, which describes how a system quality attribute fails, FMVEA proposes a *threat mode*, which describes how a security attribute of a component fails. Vulnerabilities are considered the causes for failures of security attributes. Vulnerabilities analyzed together with potential attacks form the likelihood of a threat mode. Following the system decomposition into functions, the FMVEA analysis provides a combined list of failures and threat modes together with their causes and a risk estimate. In a cyber-physical system such as a vehicle, typical functions are either processing or communication functions. As such, over time, frequently encountered failure modes for input and output have been identified [13]. Performed during system design, FMVEA provides a systematic analysis of failures and threats effects, allowing development to proceed effectively.

Because FMVEA focuses on the analysis on component functions and their interactions, it does not address the system-level functions explicitly. As a consequence, the analysis can result in a list of vulnerabilities that can be potentially exploited for each component, but may miss critical threads to the overall system.

3.3 SAHARA

Security-Aware Hazard Analysis and Risk Assessment technique has been introduced in 2015 [9] as a joint approach for performing safety and security analysis through a combination of HARA [20] for the automotive domain and STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, elevation of Priviledges) [10]. Focusing on the automotive domain, SAHARA quantifies the probability of an identified security threat and aligns its Security Level (SecL) to the ASIL (Automotive Safety Integrity Level) defined in the HARA. As in CHASSIS, functional use-cases are analyzed, but in SAHARA, the STRIDE model is used for the identification of possible security threats. In a following paper [8] Macher et al recommend using DREAD (Damage potential, Reproducibility, Exploitability, affected users, Discoverability) model instead of STRIDE for distributed systems.

3.4 STPA-Sec

System Theoretic Process Analysis - Security Method (STPA-Sec) [21] is an extension of the base STPA approach based on the STAMP (System Theoretic Accident Model and Processes) causality model. Starting from the idea that an attacker could infiltrate a system by exploring an open and undiscovered vulnerability, once the intention is there, Young and Leveson introduce the idea that the ultimate goal of security should be the assurance of maintainability of critical functions despite such intrusions [3]. Therefore, the STPA-Sec, unlike the other methodologies, does not focus on finding vulnerabilities in the individual

components, but views safety as a system property and not a component property. Focusing on enforcing safety and security concerns at the system level, hierarchical control structures are created where higher-level components constrain lower-level components. As a result, the safety and security incidents are treated as a hierarchical control problem rather than a simple chain of events.

STPA-Sec works on a specification of the system's accidents and losses, along with hazards that can potentially lead to accidents. The method performs a systematic analysis according to existing functional control diagrams. Then, a set of guide words are used to reason about causes of hazards.

4 Applying Safety and Security analysis methods on Collaborating systems

In this section, we summarize the results of comparisons resulting from application of the methods presented in the previous section on a use case from the automotive domain. A more detailed report of the method's application can be found at [6].

4.1 Use Case definition

For exemplifying the above mentioned safety and security co-analysis, we refer to a use case from the automotive domain which is general enough to ease the mapping to similar domains, such as robotics, while allowing us to be specific in the explanations. Through I2V (Infrastructure-to-Vehicle) communication, a platoon leader receives information from a RSU (Road Side Unit) about an imposed speed limit. Following a PLF (Predecessor Leader Following) [22] communication typology, the platoon leader further on sends the information to the platoon members via V2V (Vehicle-to-Vehicle) communication. In a platoon, all vehicles apply the same speed and acceleration at the same time by maintaining the closest reasonable distance. This maneuver reduces air friction and implicitly reduces the fuel consumption. A detailed description of the use case can be found at [6].

Overall, during the study and applicability of the methods, multiple differentiating factors have emerged. We group these differentiating factors in the following way:

1. ***Type of Approach***: FMVEA follows a bottom-up approach starting with system design. For performing FMVEA, a system must be divided into components such as: sensors for distance measurement, network devices, ECU controllers, etc. FMVEA maintains the focus on single-component vulnerabilities while documenting their impact on the overall system. Both SAHARA and CHASSIS, even though they do not focus on single components, are also bottom-up approaches that base the safety and security co-analysis on functional decomposition of systems. SAHARA performs a functional decomposition of systems, whereas CHASSIS performs a behavioral decomposition. STPA-Sec, on the contrary, takes a top-down approach, starting by identifying accidents and system losses. Afterwards, the analysis is conducted

with the focus on control loops between components, in order to identify if an error of control at the higher levels can potentially result in a vulnerability that can cause an accident or a loss.

2. ***Needed Artefacts***: FMVEA needs a physical decomposition diagram of a system. CHASSIS requires various forms of use cases and sequence diagrams at the start of the analysis. SAHARA requires functional decomposition diagrams of the system. STPA-Sec needs some non-standard artefacts such as: definition of accidents or losses for the system, and eventually hazard analysis.
3. ***Creativity of Analysis process***: Given the creative process of analysis, one important factor for method applicability is the way in which the analysis guidelines stimulate the thought process of a safety and security analyst. For FMVEA, the guideline directs the analyst to look at the system as a composition of components and then focus on single components for threat identification. CHASSIS provides a clear direction to focus on interaction among subsystems through the system level artefacts. STPA-Sec also clearly states that the the analysis should be focused on control loops. The only method where we did not see a clear direction on analysis is SAHARA. SAHARA provides a sophisticated mechanism for risk rating but how should one approach the system to identify those risks is not clear.
4. ***Application Phase*** The phase of development, after security analysis is completed, is a critical factor for successful risk mitigation, as the latter can become expensive and more challenging when applied in the later phases. FMVEA, CHASSIS and SAHARA can commence in the system design phase. FMVEA, due to its minimal requirements, can start earlier compared to the other two. For new system development, STPA-Sec is recommended to start at the conceptual phase as one of the earliest activities.
5. ***Generated Artefact*** FMVEA and SAHARA generate the classic quantitative tabular risk analysis summary with numbers to indicate the priority. STPA-Sec generates a functional control diagram of the system highlighting the control loops and a list of control loop issues that can result in a hazard. Among all methods CHASSIS generates the most sophisticated and graphical artefacts i.e. T-MUCs, D-MUCs and MUSDs. There is a contradicting characteristic to artefacts generated needs to be highlighted here i.e. repeatability. We think if FMVEA and SAHARA are done by multiple teams or people, the results will not vary a lot since the analysis is strictly formulated at every step. In case of CHASSIS it is partially formulated i.e. use of existing artefacts and HAZOP but still analysis is open to interpretation. STPA-Sec results of the same system by different people will probably vary most widely since it does not restrict the analysis to strict formulation.
6. ***Risk Evaluation*** All methods presented in this paper can identify security risks associated with the system. In particular, FMVEA and SAHARA provide evaluation of risks compared to each other. SAHARA integrates ASIL levels to indicate relative weighted risks. Evaluation of risks helps in prioritizing the mitigation strategies. CHASSIS does not highlight the prioritization of risks.

STPA-Sec, due to the fact that it links a hazard directly to the accidents or losses it may cause, an indirect understanding of the risk priority.

7. ***Usability*** In terms of usability and understandability, FMVEA has an advantage, being known in many industries including embedded applications in automotive domain. Both FMVEA and SAHARA methods are well-structured to be followed by any team and recommended guide words from STRIDE help stimulate the risk analysis. CHASSIS and STPA-Sec require domain knowledge in the application area and security analysis to generate an exhaustive list of risks. The artefacts generated in STPA-Sec can be complicated to understand for someone lacking understanding of control loops. Also, because this method looks beyond the system, it requires domain expertise. In CHASSIS, the artefacts like misuse cases and misuse sequence diagrams are expressed in UML, therefore an understanding of UML notations is necessary.
8. ***Efforts*** While no quantitative effort estimation has been performed during this study, in a a report from a 2018 [2], Torkildson et. al. compare efforts between FMVEA, CHASSIS and STPA-Sec for safety-security analysis of an autonomous boat quantitatively. They found out that STPA-Sec took almost double the efforts of FMVEA. CHASSIS took roughly 50% more effort than FMVEA. This is in line with our qualitative comparison with respect to the factors above i.e. STPA-Sec and CHASSIS both require a longer preparation phase and evaluation of more artefacts.

The table 1 below provides a summary of the comparison results.

Table 1: Summary of comparison

Method	FMVEA	CHASSIS	STPA-Sec	SAHARA
Approach	Bottom-Up	Bottom-Up	Top-Down	Bottom-Up
Needed Artefacts	Low Complexity: Physical Decomposition	High Complexity: US, SDs	Medium Complexity: Functional Control Diagram	Low Complexity: Functional Decomposition
Provides Risk Rating	Yes	No	No	Yes
Requires Domain Expertise	No	Yes	Yes	No
Enforces Creative Analysis	Yes	Yes	Yes	No
Repeatability of Results	High	Low	Low	High
Required Efforts	Low	Medium	Hign	Medium
Requires Special Tool Skill	No	Yes	No	No

5 Conclusions and future work

The current paper reports on comparison results between safety and security analysis methods applied on a use case example from the automotive domain. Newer security analysis methods base their approach on safety analysis methods, like FMEA and FTA, which were conceptualized 50-60 years earlier, when systems were predominantly mechanical and component-oriented. Pushed by technological developments, the digital systems we see today, including vehicles, are software intensive, tightly coupled, highly automated and connected. The platooning system we analyse in our paper is an example of such a system. The vehicles who themselves are a system of connected ECUs communicate with each other and RSUs to maintain constant longitudinal gap to improve fuel economy and traffic efficiency. The use case from the automotive domain is similar to various scenarios, including communicating and collaborating robots.

The methods reviewed in this paper represent a first attempt towards protecting a system from malicious attacks. However, according to Leveson [3], an attacker is very likely to penetrate a target system. In this case, mechanisms and methods should be devised for protecting the safe operation of a system in situations in which the attacker already made their way into the system. In the context of complex ecosystems, where actors and systems directly interact from design time to runtime, leaving developers the possibility to patch running systems with software updates, an emergent vulnerability of receiving malicious software through runtime updates becomes evident [11]. On top of this, vulnerabilities caused human stakeholders directly affect information security and therefore, human aspects of trust are becoming increasingly important future topics to be addressed by security experts.

For safeguarding a system operation at runtime, analysis methods traditionally implemented during the design need to be extended with runtime considerations as well. Mainly because the operational context of systems interacting in highly dynamic ecosystems cannot be foreseen during design time, current methods need to be extended with runtime considerations of environment characteristics that are impossible to predict.

Acknowledgment

This work was supported by the project BIECO (www.bieco.org) that received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 952702.

References

1. AUTOSAR: AUTOSAR. https://www.autosar.org/, [Online; accessed 09-June-2021]
2. Erik Nilsen, T., Li, J., Johnsen, S.O., Glomsrud, J.A.: Empirical studies of methods for safety and security co-analysis of autonomous boat. Safety and Reliability-Safe Societies in a Changing World (2018)

3. Gößling-Reisemann, S.: Resilience–preparing energy systems for the unexpected. An edited collection of authored pieces comparing, contrasting, and integrating risk and resilience with an emphasis on ways to measure resilience p. 73 (2016)
4. Ishimatsu, T., Leveson, N.G., Thomas, J.P., Fleming, C.H., Katahira, M., Miyamoto, Y., Ujiie, R., Nakao, H., Hoshino, N.: Hazard analysis of complex spacecraft using systems-theoretic process analysis. Journal of Spacecraft and Rockets **51**(2), 509–522 (2014)
5. ISO: ISO/SAE FDIS 21434 Road vehicles — Cybersecurity engineering. https://www.iso.org/standard/70918.html, [Online; accessed 08-June-2021]
6. Kar, S.R.: Report of Method's application. https://drive.google.com/drive/folders/12c3TrLUy-xLpGQmw5zfVvUECnPtxC9h2 (2021), [Online; accessed 08-June-2021]
7. Macher, G., Armengaud, E., Brenner, E., Kreiner, C.: A review of threat analysis and risk assessment methods in the automotive context. In: International Conference on Computer Safety, Reliability, and Security. pp. 130–141. Springer (2016)
8. Macher, G., Armengaud, E., Brenner, E., Kreiner, C.: Threat and risk assessment methodologies in the automotive domain. Procedia computer science **83**, 1288–1294 (2016)
9. Macher, G., Sporer, H., Berlach, R., Armengaud, E., Kreiner, C.: Sahara: A security-aware hazard and risk analysis method. In: 2015 Design, Automation Test in Europe Conference Exhibition (DATE). pp. 621–624 (2015). https://doi.org/10.7873/DATE.2015.0622
10. Microsoft: STRIDE. https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)?redirectedfrom=MSDN/ (2009), [Online; accessed 09-June-2021]
11. O'Neill, P.H.: How a $1 million plot to hack tesla failed. https://www.technologyreview.com/2020/08/28/1007752/how-a-1-million-plot-to-hack-tesla-failed/ (2020), [Online; accessed 08-June-2021]
12. Palin, R., Ward, D., Habli, I., Rivett, R.: Iso 26262 safety cases: Compliance and assurance. In: 6th IET International Conference on System Safety 2011. pp. 1–6. IET (2011)
13. Pentti, H., Atte, H.: Failure mode and effects analysis of software-based automation systems. VTT Industrial Systems, STUK-YTO-TR **190**, 190 (2002)
14. Ramaiah, B.S.M.P.S., Gokhale, A.A.: Fmea and fault tree based software safety analysis of a railroad crossing critical system. Global Journal of Computer Science and Technology (2011)
15. Raspotnig, C., Katta, V., Karpati, P., Opdahl, A.L.: Enhancing chassis: a method for combining safety and security. In: 2013 International Conference on Availability, Reliability and Security. pp. 766–773. IEEE (2013)
16. Schmittner, C., Gruber, T., Puschner, P., Schoitsch, E.: Security application of failure mode and effect analysis (fmea). In: International Conference on Computer Safety, Reliability, and Security. pp. 310–325. Springer (2014)
17. SESAMO: D4.2 Integrated Design and Evaluation Methodology. http://sesamo-project.eu/content/d42-integrated-design-and-evaluation-methodology (2014), [Online; accessed 08-June-2021]
18. SESAMO: Security and Safety Modelling. http://www.sesamo-project.eu/ (2015), [Online; accessed 08-June-2021]
19. da Silva Amorim, S., Neto, F.S.S., McGregor, J.D., de Almeida, E.S., von Flach G Chavez, C.: How has the health of software ecosystems been evaluated?: A systematic review. In: Proceedings of the 31st Brazilian Symposium on Software Engineering. pp. 14–23. ACM (2017)

20. Stolte, T., Bagschik, G., Reschka, A., Maurer, M.: Hazard analysis and risk assessment for an automated unmanned protective vehicle. In: 2017 IEEE Intelligent Vehicles Symposium (IV). pp. 1848–1855. IEEE (2017)
21. Young, W., Porada, R.: System-theoretic process analysis for security (stpa-sec): Cyber security and stpa. In: 2017 STAMP Conference (2017)
22. Zheng, Y., Li, S.E., Wang, J., Li, K., et al.: Influence of information flow topology on closed-loop stability of vehicle platoon with rigid formation. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). pp. 2094–2100. IEEE (2014)