Safety and Reliability – Safe Societies in a Changing World – Haugen et al. (Eds)
© 2018 Taylor & Francis Group, London, ISBN 978-0-8153-8682-7

Empirical studies of methods for safety and security co-analysis of autonomous boat

Erik Nilsen Torkildson, Jingyue Li & Stig Ole Johnsen
Norwegian University of Science and Technology, Norway

Jon Arne Glomsrud
DNVGL, Norway

ABSTRACT: Many autonomous systems are safety-critical, e.g., autonomous cars, boats, or aerial vehicles. Autonomous systems rely on software and communications. Security vulnerabilities of software and communication will give adversaries possibilities to attack and compromise security and safety. Therefore, when analysing safety, security should be co-analysed. In this study, we explored three safety and security co-analysis methods: Systems-Theoretic Process Analysis (STPA) plus STPA-Security Analysis (STPA-Sec), Failure Mode, Vulnerabilities and Effect Analysis (FMVEA), and Combined Harm Assessment of Safety and Security for Information Systems (CHASSIS). The purpose is to compare applicability, efficiency, and hazards identified by the different methods. An autonomous boat is used as the case study. Results of the study show that STPA plus STPA-Sec and CHASSIS can be more time consuming to use than FMVEA. However, STPA plus STPA-Sec and CHASSIS can help analysers identify more hazards of autonomous systems than FMVEA. Results of the study reveals weaknesses of each method to analyse autonomous systems with different levels of autonomy. We therefore propose possible improvements and combinations of the methods.

1 INTRODUCTION

Autonomous systems like drones, driverless cars, and autonomous boats are being developed. The key mechanism in an autonomous system is its ability to be independent of a human operator. The system manages to sustain situation awareness and decision-making capability, when an expected or unexpected event occurs. By shifting degrees of situation awareness and decision-making responsibilities from humans to the system, we can design autonomous systems with different levels of autonomy. As an example, the Society of Automotive Engineers have described 6 levels of autonomous driving (SAE, 2016) from no automation, driver assistance, partial automation, conditional automation, high automation, to full automation.

Without systematic safety/security analysis and design of autonomous systems, mishaps can happen and harm users and the environment. For example, on 24th July.2015, Fiat Chrysler Automobiles ordered recall of 1.4 million vehicles that was vulnerable to a threat of remote control and hijacking (Guzman, 2015). In 2013, Samy Kamkar demonstrated with the Parrot AR that it was possible to hijack other drones, with what he called SkyJack (Kamkar, 2013). Google self-driving cars had few accidents but was sometimes involved in a rear-end collision with human-driven cars, due to that human drivers did not anticipate actions from the autonomous system (Teoh and Kidd, 2017).

Traditionally, system safety analysis focuses on accidental component failures or software bugs. As industrial and autonomous control systems are increasingly interconnected through networks, system safety can also be compromised by security breaches. "*Although of great importance, it is not sufficient to address accidental threats (hazards) of such systems—also threats of intentional origin need to be covered* (Aven, 2007)*.*" "*Security functions are not meant to cope with physical hazards and failures; likewise, safety functions might not detect and respond to attacks that target the digital components of the system. We infer that safety and security are complementary and should be treated jointly to improve risk management* (Kriaa et al., 2015)."

Several methods have been proposed to combine safety and security analysis of industrial control systems. Some studies have empirically compared different security and safety co-analysis methods using specific systems. For example, FMVEA (Failure Mode, Vulnerabilities and Effect Analysis) and CHASSIS (Combined Harm Assessment of Safety and Security for Information Systems) were compared using an automotive cyber-physical system (Schmittner et al., 2015). The comparison focused on level of abstraction, comparability of repeated analysis, reusability of analysis artefacts, scope of analysis, suitability for risk rating, and adaptability to changing context. However, we believe that more empirical comparisons of different security and safety co-analysis methods are needed, because many methods are proposed but are not thoroughly evaluated. In addition, few studies have used autonomous systems as cases for evaluation. We have been interested in several methods including the STAMP method (STPA—System-Theoretic Process Analysis) since it has a modern system approach, looking at key control issues.

In this paper, we present an empirical study that compares three security and safety co-analysis methods using an autonomous boat that is under development as the case. The autonomous boat Revolt (www.dnvgl.com/technology-innovation/revolt/index.html) with the present design and sensor fitting is not pure autonomous yet, but a remotely operated dynamically positioned boat. This boat still misses sensors and functions for tracking other objects to be more autonomous. This study is just the first step of analysing safety and security of the autonomous boat. We will follow the development of Revolt and perform re-analysis when new functions are added. Such follow-up analyses will give us insights into different safety and security issues of autonomous systems with different levels of autonomy. Our key focus of our study was to compare *applicability, efficiency, and hazards identified by different methods.* The methods piloted and the sequence of the pilot are 1) FMVEA, 2) STPA plus STPA-Sec, and 3) CHASSIS.

Results of the study show that STPA plus STPA-Sec and CHASSIS are potentially more time consuming than FMVEA. Results also illustrate that different methods have different strengths and weaknesses for identifying different hazards. Based on results of this study, we propose to improve and combine the methods to meet the requirements of security and safety analysis of different autonomous systems.

The rest of this paper is organized as follows. Section 2 defines relevant terminologies. Section 3 introduces the state of the art of security and safety co-analysis, focusing on the three methods we evaluate. Section 4 presents our study design and results. Section 5 discusses evaluation results, and Section 6 concludes.

2 DEFINITIONS

There are many definitions of security and safety. Usually, safety is being used to describe accidental harm, while security is used to describe intentional harm. In (Firesmith, 2003), safety is defined as "*the degree to which accidental harm is prevented, reduced and properly reacted to*", and security is defined as "*the degree to which malicious harm is prevented, reduced and properly reacted to.*" In SEMA reference framework (Piètre-Cambacédès and Chaudet, 2010), safety and security are graphically mapped on a conceptual grid, which has two dimensions. The first dimension distinguishes between accidental and malicious threats. The second dimension differentiates safety and security based on origin and consequences. In the SEMA reference framework, the origin and consequence of safety is system and environment respectively. For security, the origin and consequence could be environment to system, system to environment, and system to system. In (Schmittner et al., 2016), the authors clarified the terminologies to be used for STPA plus STPA-sec analysis as follows. We follow the safety and security related definitions in (Schmittner et al., 2016) in our study.

- *Accident*: Event which causes undesired losses of life, asset damage, data, availability etc.
- *Hazard*: Dangerous system states which can lead to accidents.
- *Threat*: Potential cause of an unwanted incident, which may result in harm to a system and/or environment.
- *Vulnerability*: Weakness of an asset or control that can be exploited by one or more threats.
- *Attack*: Attempt to gain unauthorized access to or make unauthorized use of an asset.

3 STATE OF THE ART

3.1 *Security and safety co-analysis*

Many studies listed in (Kriaa et al., 2015) propose that it is necessary to consolidate the security and safety co-analysis, because security breaches can bring risks to system safety. However, the study (Eames and Moffett, 1999) identifies possible disadvantages of security and safety co-analysis "*We believe that consolidation of safety and security could reduce developers' understanding of the system being analysed, and prevent a thorough analysis of either property*." In addition, the study (Eames & Moffett, 1999) says that "*An additional danger is that a unified approach might actually hide the requirements conflicts that it aims to resolve.*" To address the possible disadvantages, it is critical to closely examine the various kinds of interdependencies between safety and security. Safety–security interactions can be classified into four categories (Piètre-Cambacédès, 2010).

- Conditional dependency: Satisfaction of safety requirements conditions security or vice-versa.

- Mutual reinforcement: Satisfaction of safety requirements or safety measures contributes to security, or vice-versa, thereby enabling resource optimization and cost reduction.
- Antagonism: When considered jointly, safety and security requirements or measures lead to conflicting situations.
- Independency: No interaction.

The security and safety co-analysis methods can generally be classified into three categories (Kriaa et al., 2015). One category is generic approach, such as FMVEA (Schmittner et al., 2014a) and Fault Tree Analysis (Kornecki and Liu, 2013). Another category is model-based graphical methods, such as CHASSIS (Raspotnig et al., 2012) and method using Bayesian Belief Networks (Kornecki et al., 2013). The third category is model-based non-graphic methods, such as STPA (Young and Leveson, 2013) and unified framework (Asare et al., 2013). Autonomous systems are often cyber-physical systems that integrate computation, networking, and physical processes. In addition, autonomous systems need to have proper situation awareness using various sensors, and need to make correct decisions based on the sensor information. Thus, we decided to evaluate one method that is relevant to cyber-physical system in each category mentioned in (Kriaa et al., 2015). We chose FMVEA, CHASSIS, and STPA plus STPA-Sec, because FMVEA and CHASSIS are shown to be applicable to automotive cyber-physical systems (Schmittner et al., 2015), and STPA plus STPA-Sec focuses strongly on software dependent systems.

3.2 *FMVEA*

FMVEA (Schmittner et al., 2014a, Schmittner et al., 2014b) is a FMEA (Failure Mode and Effect Analysis) analysis technique extended with security analysis. FMVEA is based on a three-level Data Flow Diagram (DFD). The first step of the method is to model the system and then to identify failure and threat modes of each component of the system. The failure mode covers the safety aspect, by describing the way the component could potentially fail. The threat mode covers the security aspect, describing the way the component could be potentially misused. The threat modes are based on the STRIDE model, developed by Microsoft (Microsoft, 2002). The STRIDE classification (spoofing/authentication, tampering/integrity, repudiation/non-repudiation, information disclosure/confidentiality, denial of service/availability, elevation of privilege/authorization) enables possible attacks on such components to be found. What is dependent on creating failure and threat modes is knowledge about the system. The potential risks and the effect they could have, are each related to a component (context level).

In addition to identifying vulnerability, threat modes, threat effects, and system effects, FMVEA also tries to quantify the attack probability by estimating system susceptibility and threat properties.

3.3 *CHASSIS*

CHASSIS (Raspotnig et al., 2012) defines a unified process for safety and security assessments. The process includes the use of Misuse Case (MUC) (Sindre and Opdahl, 2005) and Misuse Sequence Diagram (MUSD) (Katta et al., 2010) for visual modelling for security analysis. MUC is also used for safety assessment, but it is combined with Failure Sequence Diagram (FSD) instead of MUSD for detailed failure analysis (Raspotnig and Opdahl, 2012). As shown in Figure 1, there are three stages and 8 steps in CHASSIS. The first stage (steps 1–3) is to draw Use Case and Sequence Diagrams based on some operational and environmental descriptions of the system. In the second stage (steps 4–6), MUC diagrams are created by using a set of hazard and operability study (HAZOP) guidewords (Kletz, 1997) applied for the use cases. The MUC diagrams are then described in textual MUC templates (step 5). FSDs and MUSDs are used to refine the harm scenarios defined in the templates (step 6). When the textual Misuse cases are finished, HAZOP tables are prepared (step 7) and corresponding safety or security requirements are defined (step 8).

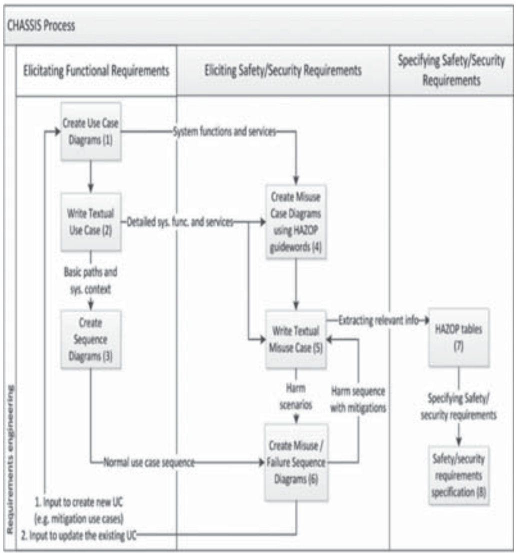
CHASSIS Process
Elicitating Functional Requirements
Eliciting Safety/Security Requirements
Specifying Safety/Security Requirements
Requirements engineering
Create Use Case Diagrams (1)
System functions and services
Write Textual Use Case (2)
Detailed sys. func. and services
Create Misuse Case Diagrams using HAZOP guidewords (4)
Basic paths and sys. context
Create Sequence Diagrams (3)
Write Textual Misuse Case (5)
Extracting relevant info
HAZOP tables (7)
Harm scenarios
Harm sequence with mitigations
Specifying Safety/ security requirements
Normal use case sequence
Create Misuse / Failure Sequence Diagrams (6)
Safety/security requirements specification (8)
1. Input to create new UC (e.g. mitigation use cases)
2. Input to update the existing UC

Figure 1. CHASSIS' unified process.

3.4 *STPA and STPA-Sec*

STPA-Sec (Young and Leveson, 2013, Young and Leveson, 2014) extends STPA, which is a safety analysis method (System-Theoretic Process Analysis) (John P., 2013, Leveson, 2012). The extension is to includes security analysis. STPA-Sec "*Shifts the focus of the security analysis away from threats as the proximate cause of losses and focuses instead on the broader system structure that allowed the system to enter a vulnerable system state that the threat exploits to produce the disruption leading to the loss (Young and Leveson, 2013)*."

The main steps of STPA plus STPA-Sec are:

- Identifying what essential services and functions must be protected or what represents an unacceptable loss.
- Identifying system hazards and constraints.
- Drawing the system control structure, physical hardware and network structure, and identifying unsafe control actions.
- Determining the potential causes of the unsafe control actions. The potential causes could be security vulnerability and threats. To facilitate the security analysis, some guide words like tampered feedback, injection of manipulated control algorithm, and intentional congestion of feedback path, are added (Schmittner et al., 2016).

Compared to other security analysis methods, STPA-Sec does not focus on countermeasures that should be taken. STPA-Sec focuses mainly on identifying those scenarios that could lead to losses.

4 STUDY DESIGN AND RESULTS

4.1 *Scope: Autonomous boat*

The autonomous boat Revolt shown in Figure 2 was made by Stadt Towing Tank (STT), on a mission from DNVGL in 2014. The model is a 1:20 scale model of the concept ship. The model ship has a length of 3 meters and weighs 257 kg.

Although Revolt is still under development and is not a fully autonomous boat, we still want to use it as a case since it gives us the opportunity to explore hazard and threats of two main issues i.e. 1) Safety and security of autonomous steering of the ship (i.e. losing control; ship damaged/ destroyed) and 2) security of data-communication between onshore and offshore (sensitive data compromised).

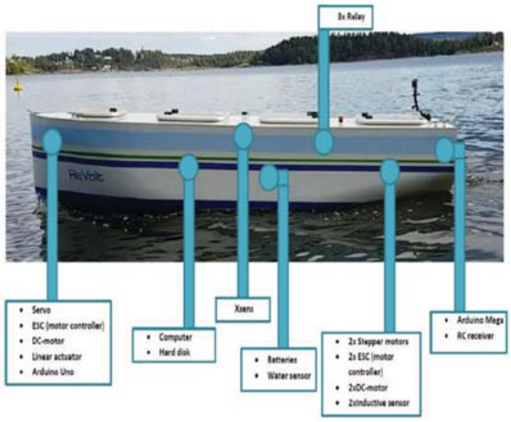

Figure 2. Overview of Revolt and its components.

4.2 *Security and safety co-analysis using FMVEA*

The FMVEA analysis focuses on the embedded computer. The attack surface is the highest for the embedded computer in the Revolt, since other components in some way are connected to it. Microcontrollers are connected to (and controlled by) the embedded computer via USB. Analogue components (water sensor etc.) are connected to the microcontrollers.

To perform the FMVEA analysis, we fill in the table as proposed in (Schmittner et al., 2014a). The table includes columns for qualitative safety and security analysis, such as component, failure mode, threat mode, failure effect, threat effect, system status, system effect. The table also includes columns, such as severity, system susceptibility, treat properties, attack/failure probabilities, and risks, for quantitative analysis and for ranking the hazards.

4.3 *Security and safety co-analysis using STPA plus STPA-Sec*

When performing STPA plus STPA-Sec analysis, we start with the following unacceptable losses/ accidents and safety constraints.

- Collision with vessels, objects, humans/mammals, structures, grounding
- Fire or explosion
- Foundering (sinking, failing or plunging)
- Loss of cargo
- Loss of mission objectives
- Loss of information

Then, we read the network structure and the control structure documents of the boat to identify unsafe control actions. We follow the systematic method proposed in (John P., 2013) and enumerate full combinations of possible values of process variables and evaluate where control actions can be unsafe if the control action is given, is not given, is given too early or too late, too large or too small value. The control actions (CAs) we analyse include:

- CA1: Control the position of the vessel
- CA2: Control the speed of the vessel
- CA3: Control the course of the vessel
- CA4: Control the access to the vessels system

After identifying the Unsafe Control Actions (UCA), the last step of the analysis is to identify possible causal factors of the UCA, including possible security breaches that can lead to the UCA. In this last step, STPA-Sec analysis is applied by using the guide words proposed in (Schmittner et al., 2016).

4.4 *Security and safety co-analysis using CHASSIS*

To perform CHASSIS analysis, we first identify use cases and draw use case diagrams. The use case we focus on is "operating and monitoring Revolt remotely through the Revolt Intelligent System (RIS)". Then we make security and safety misuse case through using the HAZOP keywords proposed in (Schmittner et al., 2015). Examples of the safety and security misuse cases are shown in Figure 3.

4.5 *Comparisons of effort spent on co-analysis*

The inputs to the methods are very different. FMVEA analysis focuses on components. STPA plus STPA-Sec analysis focuses on control actions. CHASSIS analysis focuses on use cases. Thus, it is difficult to have direct comparisons of the effort spent on applying the methods. However, by analysing the hours spent on each activity shown Table 1, we can still observe that STPA plus STPA-Sec and CHASSIS can be more time-consuming than FMVEA, because more activities are included and each activity requires more effort.

4.6 *Comparisons of safety hazards identified*

Like comparisons of effort, it is difficult to perform direct comparisons of safety issues identified by using different methods, because the methods have different inputs. However, through comparing safety issues identify by each method, we can observe strengths and weaknesses of each method. FMVEA helps us identify mostly the hazards that are related to single component failure, e.g., communication connection is lost or updates fails. The input of FMVEA does not require as many inputs as the two other methods. It requires only a list of components of the system and how they are connected. This is an advantage. However, it may also be a restriction for early analysis, because early system development might not have a system design.

Compared to FMVEA, STPA plus STPA-Sec method helps us identify more hazards that are related to interactions between different components or actors. STPA is a top down approach that

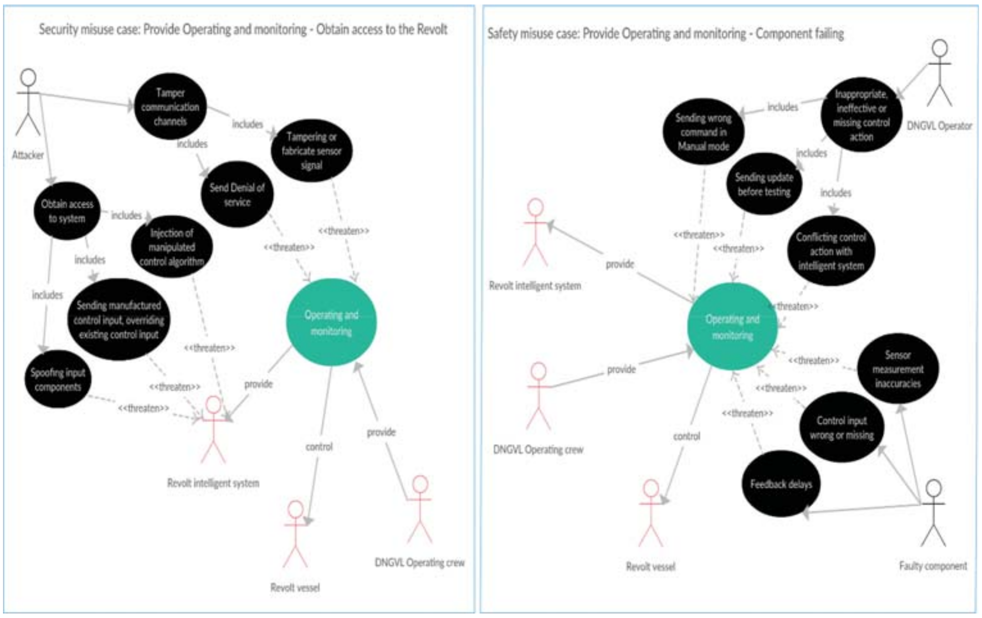
Security misuse case: Provide Operating and monitoring - Obtain access to the Revolt
Attacker
Tamper communication channels
includes
Tampering or fabricate sensor signal
includes
Send Denial of service
Obtain access to system
includes
Injection of manipulated control algorithm
includes
includes
Sending manufactured control input, overriding existing control input
Spoofing input components
<<threaten>>
<<threaten>>
<<threaten>>
<<threaten>>
<<threaten>>
Operating and monitoring
provide
Revolt intelligent system
control
provide
Revolt vessel
DNGVL Operating crew
Safety misuse case: Provide Operating and monitoring - Component failing
Sending wrong command in Manual mode
includes
Inappropriate, ineffective or missing control action
DNGVL Operator
includes
Sending update before testing
includes
Conflicting control action with intelligent system
<<threaten>>
<<threaten>>
<<threaten>>
Revolt intelligent system
provide
Operating and monitoring
provide
DNGVL Operating crew
<<threaten>>
Sensor measurement inaccuracies
<<threaten>>
Control input wrong or missing
<<threaten>>
control
Feedback delays
Revolt vessel
Faulty component

Figure 3. Examples of security and safety misuse cases.

Table 1. Effort spent on each method.

FMVEA		STPA and STPA-Sec		CHASSIS	
Activities	hr.	Activities	hr.	Activities	hr.
System level analysis	5	Define unacceptable losses	7	Elicitation functions and services	4
Selection of component	3	Identify hazards and safety constraints	5	Use case diagram	9
Identify functions of component	10	Create functional control structure	30	Safety misuse case diagram	9
Failure Mode, Vulnerabilities and Effect Analysis	27	Identify hazardous control actions	20	Security misuse case diagram	9
Risk assessment	10	Identify causal factors and scenarios	20	Final misuse case with mitigations	15
		Identify mitigations	23	Misuse Sequence Diagram	8
				Failure Sequence Diagram	8
				Fill in HAZOP table	20,5

looks at the operative picture and identifies unsafe system operation. STPA analysis covers not only the physical system, but also human operators and actors. One hazard example identified by STPA is "setting route for shipment and launch position when the shipping dock has not permitting the action, because other ships are dispatching at the same time". STPA can identify such hazards due to its use of process model variables to identify hazardous control actions, which generates scenarios that otherwise would be omitted. Another advantage of STPA is that it does not assume or need the fully designed observability in the system from the beginning, and this can possibly be achieved through several iterations. We usually find out some UCA based on the preliminary design of the system. When we explore casual factors of UCA, we may find new constraints or requirements for observability and control to handle the identified accident causes, or new need to obtain proof that the accident causes will not practically occur. However, the challenge of STPA plus STPA-Sec analysis (John P., 2013) is that it relies heavily on enumerating process control variables. If many process control variables are present, the analysis can be time consuming.

Compared to FMVEA and STPA, the strength of CHASSIS is that it helps us find hazards that are related to operation sequences. One example hazard identified by using CHASSIS is "the operator performs operations on the Revolt before having done security and safety procedures, and the Revolts components are having feedback delays and commands are executed too late". The weakness of CHASSIS is that it relies more on expert judgement than FMVEA and STPA. As observed in (Schmittner et al., 2015), the possible risk could be that "*if a CHASSIS analysis is repeated by a new group, due to the differences in the experts, new viewpoints can be introduced that change the results*." A restriction of CHASSIS we identify is that its starting point is the use case. If the use case is too broad, steps that follows in the process might be difficulty to perform.

4.7 *Comparisons of security issues identified*

FMVEA security analysis uses STRIDE classification. The identified security threats are limited to threat targeted at single component, e.g., wireless connection is targeted to jamming. When using FMVEA, the safety and security analysis can be done independently. Thus, safety–security interactions may be overlooked.

CHASSIS identifies threats and vulnerability using misuse cases. The hazards identified by CHASSIS are mostly related to operation and use of the system, e.g., the communication system might have vulnerabilities that could lead to modification of system files. By integrating security misuse cases and safety misuse cases, it is possible to analyse safety–security interactions. However, like the safety analysis, the possible weakness of CHASSIS is that it relies heavily on expert knowledge. Thus, the analysis results may not be replicable.

The security analysis of STPA-Sec focuses on identifying security vulnerabilities that may lead to unsafe control actions. For example, providing CA2 (control the speed of the vessel) too late from shore to the boat when the WIFI connection is jammed. Comparing to FMVEA, the strength of STPA plus STPA-Sec is that it focuses more on safety–security interactions. However, the limitation of the STPA-Sec is that the security analysis focuses mainly on vulnerability that can be the casual factors for safety hazards. The security vulnerabilities, which may lead to information leakage or privacy issues, but will not lead to safety hazards, may be overlooked. The study (Schmittner et al., 2016) proposes to enhance STPA-Sec with more focus on losses related to confidentiality. In our study, we list "loss of information" as an accident and find out some threats that can lead to this loss. However, STPA plus STPA-Sec method use enumeration of process control variables to identify possible information loss. If certain security vulnerabilities, e.g. improper encryption of stored data, are not reflected directly in existing process control variables, the vulnerabilities may not be identified in the analysis. Thus, we believe that integrating STPA-Sec with more security oriented analysis methods, e.g., misuse cases or threat modelling, can be beneficial.

5 DISCUSSIONS

5.1 *Comparison with related studies*

The study (Schmittner et al., 2015) compared FMVEA and CHASSIS. Our study included STPA and STPA-Sec in the comparison.

- Level of abstraction: CHASSIS is a quite high-level approach. It can be applied in early requirement and concept phase, when a system is not clearly defined and little information is known. In contrast, FMVEA needs at least a list of system elements and connections between the elements to generate meaningful results (Schmittner et al., 2015). STPA plus STPA-Sec requires information of the hardware, the network nodes, the network input/output lists to identify process control variables and unsafe control actions.
- Replicable analysis results: CHASSIS depends more on expert knowledge. In contrast, FMVEA and STPA will more likely provide comparable results, even if the analysis is performed by different persons.
- Reusability of analysis artefacts: All three methods use guidewords. FMVEA uses failure modes and STRIDE classification. STPA plus STPA-Sec uses guide words proposed in (Schmittner et al., 2016). CHASSIS uses HAZOP keywords. In all three methods, the quality and completeness of the keywords will strongly influence the quality of the analysis.
- Scope of analysis: FMVEA and STPA plus STPA-Sec depend to a higher degree on the accuracy of the system model and control structure. For CHASSIS, "*it is possible to expand the consideration of risk scenarios which do not arise directly from the system model* (Schmittner et al., 2015)."
- Suitability for a risk rating: FMVEA targets at rating the risks. STPA plus STPA-Sec and CHASSIS focus mostly on generating a list of possible safety and security issues rather than rating them. A possible combination of the method is to perform STPA plus STPA-sec or CHASSIS analysis to identify hazards and then use FMVEA for quantitative comparisons of certain hazards.
- Adaptability to changing context: It is easier for CHASSIS to consider different usage scenarios and changing environment than FMVEA, because CHASSIS is less formal and focuses on high level analysis. STPA plus STPA-Sec and FEMVA analyses results need to be updated when the system design changes.

5.2 *Applicability of the methods for analysing autonomous systems*

Autonomous systems have different levels of autonomy. Based on our observations of strengths and weakness of the three methods, we propose applying different methods for analysing systems with different levels of autonomy.

- For systems with high automation, STPA plus STPA-Sec may be more applicable than FMVEA to analyse interactions between systems, and interactions between systems and environment.
- For systems with many sensors, STPA plus STPA-Sec may be more applicable than CHASSIS. CHASSIS focuses on sequential messages. In contrast, STPA deals with fusions of sensor messages that come at the same time better. However, STPA plus STPA-Sec also needs to be improved. The current STPA proposed in (John P., 2013) is limited to analyse single control action. For many cyber-physical systems and autonomous systems like autonomous boat, some control actions are mutually dependent and might be issued in pairs. For example, in emergency cases, the boat needs to change course and slow down at the same time to avoid collision. Our solution for analysis mutually dependent control actions is to add the control action as a process control variable of another control action, if another control action has dependency with it. For example, in the table to analyse control action CA3 (i.e. control the course), the CA2 (i.e. Control the speed) is added as process control variables with values "speed up" and "slow down".
- For autonomous system with high level intelligence and learning capability, none of the three methods will be sufficient. AI will make it harder to review the system due to its increasing "black

box" and "black code" nature and its learning capability. For those systems, STPA plus STPA-sec or CHASSIS analysis may outperform FMVEA, because the operational level is the same regardless of system implementation. STPA and CHASSIS are good at analysing the operational safety with the system interaction. For autonomous systems with learning capability, however, it is necessary to have continuous verification along with the learning.

5.3 *Limitations of the study*

One main limitation of this study is that the safety and security hazards identified by this study may not be complete. It is because the completeness relies much on the domain knowledge and the guide words. However, the purpose of the study is to compare the three methods rather than to identify all hazards of the system. We believe that, even if other researchers identify slightly more security and safety hazards than us or identify different hazards from Revolt, our observations of the main differences of the three methods are still valid.

6 CONCLUSIONS AND FUTURE WORK

Many security and safety co-analysis methods have been proposed from academia and industry. However, few empirical studies have been performed to compare and evaluate the methods. In this study, we have evaluated three methods using an autonomous boat, called Revolt, as a case study. Results of the study show advantages and disadvantages of each method. Our future study is to extend and strengthen existing methods to analyse safety and security issues of intelligent and complex control actions of autonomous systems. In addition, we need to check the validity of the method, based on observing performance and incidents of the Revolt system.

ACKNOWLEDGEMENT

This work is supported by the SAREPTA (Safety, autonomy, remote control and operations of industrial transport systems) project, which is financed by Norwegian Research Council with Grant No. 267860.

REFERENCES

Asare, et al.. (2013) FSTPA-I: a formal approach to hazard identification via system theoretic process analysis. P*roceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems.* Philadelphia, Pennsylvania, ACM.

Aven, T. (2007) A unified framework for risk and vulnerability analysis covering both safety and security. *Reliability Engineering & System Safety,* 92, 745–754.

Eames, D.P. & Moffett, J. (1999) The Integration of Safety and Security Requirements. In Felici, M. & Kanoun, K. (Eds.) Computer Safety, Reliability and Security: 18th International Conference, SAFECOMP'99 Toulouse, France, September 27–29, 1999 Proceedings. Berlin, Heidelberg, Springer Berlin Heidelberg.

Firesmith, D. (2003) Common Concepts Underlying Safety, Security, and Survivability Engineering. Carnegie Mellon University.

Guzman, Z. (2015) Hackers remotely kill Jeep's engine on highway.

John P., I., Thomas (2013) Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis. Massachusetts Institute of Technology.

Kamkar, S. (2013) SkyJack, http://samy.pl/skyjack/.

Katta, V., Karpati, P., Opdahl, A.L., Raspotnig, C. & Sindre, G. (2010) Comparing Two Techniques for Intrusion Visualization. In Van Bommel, P., Hoppenbrouwers, S., Overbeek, S., Proper, E. & Barjis, J. (Eds.) The Practice of Enterprise Modeling: Third IFIP WG 8.1 Working Conference, PoEM 2010, Delft, The Netherlands, November 9–10, 2010. Proceedings. Berlin, Heidelberg, Springer Berlin Heidelberg.

Kletz, T.A. (1997) Hazop—past and future. R*eliability Engineering & System Safety,* 55, 263–266.

Kornecki, A. & Liu, M. (2013) Fault Tree Analysis for Safety/Security Verification in Aviation Software. Electronics, 2, 41.

Kornecki, A.J. et al. (2013) Studying interrelationships of safety and security for software assurance in cyber-physical systems: Approach based on bayesian belief networks. 2013 Federated Conference on Computer Science and Information Systems.

Kriaa, S. et al. (2015) A survey of approaches combining safety and security for industrial control systems. *Reliability Engineering & System Safety*, 139, 156–178.

Leveson, N.G. (2012) *Engineering a Safer World: Systems Thinking Applied to Safety,* MIT Press.

MICROSOFT (2002) The STRIDE Threat Model.

Piètre-Cambacédès, L. & Chaudet, C. (2010) The SEMA referential framework: Avoiding ambiguities in the terms "security" and "safety". *International Journal of Critical Infrastructure Protection*, 3, 55–66.

Piètre-Cambacédès, L. (2010) Des relations entre sûreté et sécurité. (The relationships between safety and security).

Raspotnig, C. & Opdahl, A. (2012) Supporting Failure Mode and Effect Analysis: A Case Study with Failure Sequence Diagrams. In Regnell, B. & Damian, D. (Eds.) Requirements Engineering: Foundation for Software Quality: 18th International Working Conference, REFSQ 2012, Essen, Germany, March 19–22, 2012. Proceedings. Berlin, Heidelberg, Springer Berlin Heidelberg.

Raspotnig, C. et al. (2012) A Combined Process for Elicitation and Analysis of Safety and Security Requirements. In Bider, I., Halpin, T., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Soffer, P. & Wrycza, S. (Eds.) *Enterprise, Business-Process and Information Systems Modeling.* Berlin, Heidelberg, Springer Berlin Heidelberg.

SAE (2016) SAE International standard "J3016: Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems.".

Schmittner, C. et al. (2014a) Security Application of Failure Mode and Effect Analysis (FMEA). In Bondavalli, A. & Di Giandomenico, F. (Eds.) Computer Safety, Reliability, and Security: 33rd International Conference, SAFECOMP 2014, Florence, Italy, September 10–12, 2014. Proceedings. Cham, Springer International Publishing.

Schmittner, C. et al. (2014b) FMVEA for Safety and Security Analysis of Intelligent and Cooperative Vehicles. In Bondavalli, A., Ceccarelli, A. & Ortmeier, F. (Eds.) Computer Safety, Reliability, and Security: SAFECOMP 2014 Workshops. Florence, Italy, September 8–9, 2014. Proceedings. Cham, Springer International Publishing.

Schmittner, C. et al. (2015) A Case Study of FMVEA and CHASSIS as Safety and Security Co-Analysis Method for Automotive Cyber-physical Systems. Proceedings of the 1st ACM Workshop on Cyber-Physical System Security. Singapore, Republic of Singapore, ACM.

Schmittner, C. et al. (2016) Limitation and Improvement of STPA-Sec for Safety and Security Co-analysis.

Sindre, G. & Opdahl, A.L. (2005) Eliciting security requirements with misuse cases. *Requirements Engineering*, 10, 34–44.

Teoh, E.R. & Kidd, D.G. (2017) Rage against the machine? Google's self-driving cars versus human drivers. *Journal of Safety Research,* 63, 57–60.

Young, W. & Leveson, N. (2013) Systems thinking for safety and security. *Proceedings of the 29th Annual Computer Security Applications Conference.* New Orleans, Louisiana, USA, ACM.

Young, W. & Leveson, N.G. (2014) An integrated approach to safety and security based on systems theory. Commun. ACM, 57, 31–35.