INSpIRA: INtegrating Security Into Risk Assessment

Doctoral Project Paper

Jürgen Dobaj
Graz University of Technology
Graz, Austria
juergen.dobaj@tugraz.at

ABSTRACT

Today's software and hardware technologies enable the expansion of Cyber-Physical Systems (CPSs) into the realms of mobility (car2x, autonomous driving), energy (power plants, smart grid) and healthcare (health monitoring), paving the way into a highly interlaced world. However, this also dramatically broadens the threat landscape for potential attacks on CPSs. The malfunction of these CPSs could threaten human life, cause environmental damage and major financial loss [5, 9, 16]. This drives the need for comprehensive methods that support the cross-domain design, development and implementation of safe and secure systems [3, 4]. In order to tackle these challenges, this paper proposes a method called INSpIRA, a method for INtegrating Security Into Risk Assessment, including a toolchain implementing the method. The envisioned method is supposed to be a holistic approach that supports the efficient cross-domain design, development, implementation and maintenance of dependable CPSs, where security and safety are a critical aspect that requires an in-depth risk assessment.

CCS CONCEPTS

• **Security and privacy** → *Embedded systems security*; • **Hardware** → *Safety critical systems*; • **Software and its engineering** → *Risk management*;

KEYWORDS

Dependability, Safety, Security, Risk Assessment, Self-Adaptive System, Model Based Development, Industrial Control Systems, ICS

ACM Reference Format:
Jürgen Dobaj. 2018. INSpIRA: INtegrating Security Into Risk Assessment: Doctoral Project Paper. In *SEAMS '18: SEAMS '18: 13th International Symposium on Software Engineering for Adaptive and Self-Managing Systems , May 28–29, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3194133.3194159

1 INTRODUCTION

CPSs expand into the mobility (car2x, autonomous driving), energy (power plants, smart grid) and healthcare (health monitoring) sectors, which paves the way into a highly connected world. However, a highly connected world increases the risk of potential attacks on such CPSs. This drives the need for comprehensive methods that support the cross-domain development of safe and secure systems, because the malfunction of such CPSs could threaten human life, cause environmental damage and/or major financial loss. The Threat Horizon 2017 report lists "death from disruption to digital services" [5] as one of these threats, especially in the medical and mobility domain. Additionally, the newest Threat Horizon 2019 report claims that the cause of service disruption may come "from an over reliance on fragile connectivity" [9]. In short, the threatening of human life and other assets by intentional attacks on interlaced CPSs encourages the development of integrated safety and security methods for risk assessment.

This paper proposes the INSpIRA method and toolchain. INSpIRA is supposed to combine and extend established risk and security analyses methods to automatically generate test cases and resilience mechanisms that are both implemented by a generic execution engine. The test engine encourages the continuous evaluation and improvement of the CPS, while the safety and security engine can be integrated into the CPS to extend or completely replace runtime safety and security mechanisms. This approach is intended to make the development process more efficient, less tedious and less error-prone. The proposed architecture of the generic execution engine uses self-adaptive loops (MAPE-K [11]) to achieve its goals.

2 RELATED WORK

Cherdantseva et al. [3] reviewed twenty-four security risk assessment methods for the development of Supervisory Control and Data Acquisition (SCADA) systems. They pointed out that tool support would be especially beneficial to automatically generate and analyze risk models, recommend countermeasures, or even trigger them as a response to undesired events. Additionally these tools could ease the evaluation of different methods on real and more complex use cases in a less tedious way, if the risk assessment processes was at least partially automated. However, out of the twenty-four evaluated papers only seven discussed tool support and neither tool architectures nor user interfaces were demonstrated. Cherdantseva et al. also concluded that future risk assessment tools may consider the capturing and formalisation of expert opinion and the evaluation and validation of the risk assessment process [3].

Chockalingam et al. [4] conducted a survey to identify the key characteristics and applications of integrated safety and security risk assessment methods to provide a base for future investigations and the development of more effective risk assessment methods and tools. Based on their inclusion criteria they evaluated seven methods [2, 13, 14, 17–20] and identified that none of the methods take into account runtime system information to perform dynamic risk assessment.

SEAMS '18, May 28–29, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5715-9/18/05.
https://doi.org/10.1145/3194133.3194159

Summing up, despite the fact that a large number of methods and tools for integrated safety and security risk assessment and management exist, there is still room for further research and tool improvements.

3 INSPIRA

As pinpointed in the previous section, there is a need for comprehensive methods and tools that support the inter-domain development of safe and secure systems. In order to make the next step in integrated security and safety assessment, this paper proposes a method and toolchain for INtegrating Security Into Risk Assessment (INSpIRA). The envisioned method is supposed to be a holistic approach that supports the efficient cross-domain design, development, implementation and maintenance of safety and security relevant CPSs. The INSpIRA workflow is separated into two phases, the system specification phase and the runtime phase. Based on these phases this section explains the proposed toolchain . The phases and an example are shown in Figure 1. The example shown in Figure 1(b) focuses on INSpIRA's structure and system design, while the example described in section 3.3 is based on a concrete use case and discusses potential methods that can be used to construct a knowledge base (see Figure 2).

3.1 System Specification Phase

Set of Safety and Security Analysis Methods: The INSpIRA method allows to combine different risk and security analyses methods to identify the safety and security requirements of the System under Development (SuD). Depending on the given use case or other development constraints the user can define which set of analysis methods the engineering toolchain should use.

The Engineering Toolchain: supports the user by applying the previously selected methods. The knowledge generated during method execution should be human and machine readable. Therefore the toolchain organizes the knowledge in tree and net structures, similar to the APIS risk analysis tool [10]. The tree and net structures are divided into five main groups: (1) Requirements Tree, (2) Structure Tree, (3) Function Net, (4) Failure Net, and (5) Recovery Strategies. The groups and structures are interconnected, as exemplarily shown in Figure 1(b). Depending on the applied method different groups in the knowledge base are affected and modified. The following steps explain the engineering toolchain's workflow (see Figure 1(a)) and the relationship between the five main groups:

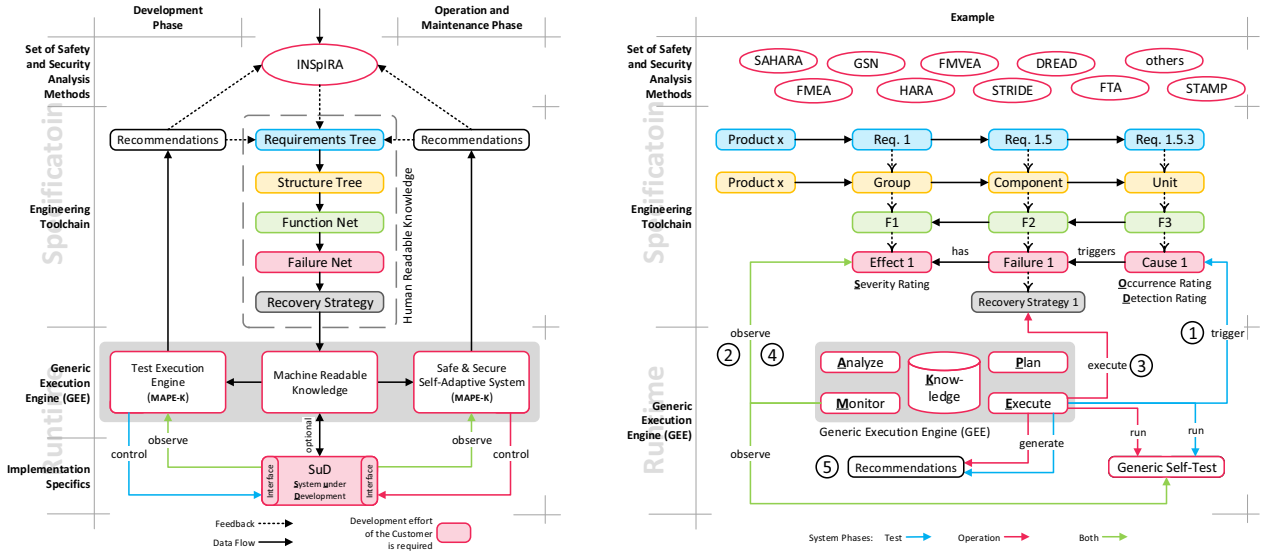
(1) The development of a system generally starts with the concept phase where the system requirements are defined mainly in a textual manner, which is stored in the **Requirements Tree**. In order to obtain a more technical and traceable requirements specification a systematic approach based on the Failure Mode, Vulnerability and Effect Analyzes (FMVEA) [18] will be used. The FMVEA requires that the system's structure, functions, potential failures, vulnerabilities and countermeasures have to be identified. This is explained in the following by the steps 2, 3 and 4.
(2) The FMVEA's first step is the division of the system into groups, components and units, which is stored in the **Structure Tree**. The structure tree might already contain information obtained by methods applied in the previous step like block diagrams, network models or threat models. All elements in the structure tree are linked to their describing element in the requirements tree.
(3) As each element of the structure tree implements one or more specific functions, the structure tree is linked to at least one element in the **Function Net**. The elements in the function net describe the provided functionality or service and contain a technical description of the structure tree element (e.g., flowchart, interface definition, implementation).
(4) Per definition a function could fail or could be the target of an attack. Therefore, the FMVEA requires the analysis of potential failures and vulnerabilities by constructing Cause-Mode-Effect or Vulnerability-Thread-Effect relations that are stored in the **Failure Net**.
(5) The INSpIRA method extends the FMVEA and additionally requires the definition of **Recovery Strategies** that are linked to the failure modes defined in the failure net. A *Recovery Strategy* describes the necessary actions to either enter a safe state or to recover from a failure, if possible.

Execution Engine: The *Human Readable Knowledge* constructed in the previous steps is then used to generate the *Machine Readable Knowledge* for the *Generic Execution Engine (GEE)*. The GEE is implemented as a self-adaptive system and uses MAPE-K loops [11] to integrate the security and safety knowledge for testing the SuD, and to assure that the defined safety and security goals are met during operation. The test results should provide evidence about the SuD's safety and security properties. If discrepancies are detected, the GEE generates a notification and tries to recommend potential fixes.

Implementation: The SuD, of course, has to be implemented based on the previously engineered requirements specification. The INSpIRA method requires the implementation of an *Interface* for the interaction between the GEE and the SuD, which imposes an additional development effort. However, the GEE is planned to already implement relevant parts of the required testing mechanisms and system functionality, which significantly reduces the overall development effort. The *Interface* also encapsulates implementation specifics, which allows to use a digital twin [7] of the SuD during development. This digital twin could be an arbitrary implementation enabling the support for simulation-based verification [21].

3.2 Runtime Phase

In its runtime phase INSpIRA tests and assures the safety and security properties of a dependable system using auto-generated mechanisms based on the knowledge generated during the system specification phase. This knowledge contains the SuD's structure, functions, potential failures, vulnerabilities and recovery strategies. The knowledge is supposed to be used for test case generation in the development and operation phase (see Figure 1(a)), but in the future it will also be used to implement the SuD's safety and



(a) The INSpIRA Workflow showing the development and operation phases.

(b) An example including the GEE's test and operation data flow.

Figure 1: The INSpIRA workflow and an example.

security mechanisms for the operation phase, which is exemplified in Figure 1(b).

Development Phase: In the *development phase* the FMVEA requires the definition of Cause-Mode-Effect relations for all identified failure modes. The GEE uses this information to generate test cases and to run its test execution engine. Each test case then (1) triggers a cause that triggers a failure. The SuD, of course, must be able to detect that failure and should react accordingly. The GEE (2) observes the reactions of the SuD and based on this observations the GEE may decide if the SuD works correctly. During this process the GEE also observes the expected effects. If after the cause triggering no failure can be detected, this is a hint towards a potential design or implementation failure. Then the GEE (5) generates a report and notifies a development engineer to reevaluate the SuD. This approach encourages to constantly evaluate and improve the SuD and its requirements.

Operation and Maintenance Phase: In the *operation and maintenance phase* the triggering of failures is deactivated, because forcing the system into a failure mode would provide a serious attack vector and safety issue. Instead, the GEE runs auto-generated self-tests at system startup and during operation the GEE acts as a self-adaptive system executing safety and security mechanisms. Therefore the GEE constantly observes the SuD's behavior. Once the GEE observes effects that indicate a failure mode, the GEE can execute a corresponding *Recovery Strategy* . If the GEE detects an undesired behavior that is not documented by a Cause-Mode-Effect relation, the GEE should also notify the engineering team to request improvement or maintenance. If the detection of failures and the recovery can be established within an acceptable timespan or even in real-time, then the GEE can replace parts or the entire safety and security mechanisms, otherwise implemented in the SuD. If the real-time reaction is not possible, the GEE can still be used to implement resilience mechanisms, such as re-routing, hot standby device management, fault recovery and maintenance cycle planing [8].

Self-Evaluation: If the GEE can be used to establish both, the generation of test cases and the implementation of safety and resilience mechanisms for the SuD, then the GEE would be able to test itself. In order to test itself two instances of the GEE must be launched. One instance runs the *Test Execution Engine* and the other instance executes the *Safe & Secure Self-Adaptive System*. This scenario is exemplified in Figure 1(b), where the *Test Execution Engine* (1) triggers a cause, which should result in a failure mode and observable effects. (2) The *Safe & Secure Self-Adaptive System* observes these effects and (3) executes a *Recovery Strategy* after detection. (4) The *Test Execution Engine* again observes the SuD's reactions and finally (5) generates a report and recommendations.

3.3 Applications

The INSpIRA approach is supposed to be used as holistic approach that supports the efficient cross-domain design, development, implementation and maintenance of dependable CPSs, where security and safety are a critical aspect that requires an in-depth risk assessment. Examples of such systems are modern road vehicles (assisted and autonomous driving), CPS road networks (car2x), power plants, smart grids or healthcare devices. These systems operate in different domains, under different environmental conditions and technical or legislation constraints. Thus, diverse approaches and methods tailored for dependable system development in a specif domain exist. This diversity addresses INSpIRA by providing the possibility to choose the set of methods that should be used for

system development. The example in Figure 2 shows methods supposed to be suitable for the development of CPS road networks. The system specification starts with brainstorming resulting in textual requirements. To cover potentially relevant environmental influences of a large scale CPS road network the System-Theoretic Accident Models and Processes (STAMP) method [12] can be used in the initial concept phase to construct a high level large scale system model. In the following the Security-Aware Hazard And Risk Analysis (SAHARA) method [13] is applied to assign either a Safety Integrity Level (SIL) or a Security Level (SecL) to the identified requirements. The subsequent FMVEA [19] refines the system structure, describes the system functionality and identifies potential failures of safety relevant functions. As a last step in the system specification phase the recovery strategies for failure modes are defined. The constructed knowledge base can then be used as input to configure the runtime system for testing, operation, resilience and maintenance.

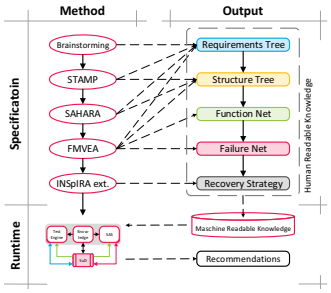


Figure 2: Concrete example of the INSpIRA workflow.

4 DISCUSSION

The tool supported INSpIRA workflow accumulates the system's requirements, structure and function into a single knowledge base. This encourages experts from different domains, departments and even companies to collaboratively design and develop safe and secure systems. For the collaborative development across departments or companies client capability will be required to protect intellectual property. Also the integration of different methods is a challenging problem and worth being discussed in future work.

The proposed toolchain tries to combine the benefits of model based design, development and testing [1] into a single workflow, and the encapsulation of the SuD via interfaces allows to use digital twins during the product development. This reduces the management overhead and makes the development process more efficient, less tedious and less error-prone. Integrating already established model based methods into the INSpIRA workflow is worth considering.

Arguing about the safety and security properties of a system is often a hard problem companies are faced with [6]. Companies have to convince their customers or the legislation that the developed systems achieve the required safety and security goals, which may also include the certification of the entire development process. The use of different environments during product development, testing and operation complicates this certification. The envisioned INSpIRA toolchain can not only be used for testing and implementing safety and security requirements, it also establishes links between the requirements, its implementation, its testing and the operation of the system. This allows to track and argue about the fulfillment of the desired safety and security goals. Once the establishment of the links is certified, the INSpIRA method can be used as a holistic method to build products, all having a strong safety and security argument.

The toolchain uses the same information source and technology for testing, operation and self-validation. This reduces the development and integration effort, but might result in systems having the same limitations regarding error detection and controllability. This could be addressed by adding additional test cases, or by using different information sources or technologies. There is a huge potential of using self-adaptive systems for industrial automation and controlling CPS [8]. However, the applicability of the INSpIRA approach must be evaluated for domains where it is likely that the system's uncertainty is not controllable.

5 DOCTORAL PROJECT PLAN

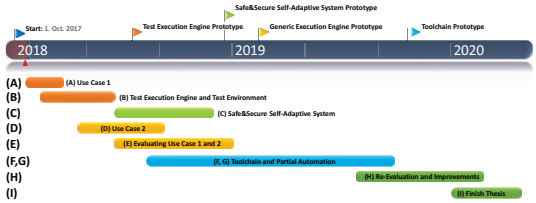


Figure 3: Timeline of the Doctoral Project Plan.

This section outlines the research hypotheses and states the open questions that are planned to be answered during the dissertation research. A timeline for the completion is shown in Figure 3. At the point of writing no results or achievements can be presented, besides the proposed INSpIRA method.

The previous section proposes INSpIRA, a method for INtegrating Security Into Risk Assessment that supports the efficient cross-domain design, development, implementation and maintenance of safety and security relevant CPSs. The systems developed using the INSpIRA method are claimed to have a strong safety and security argument. In order to give evidence of these properties it is planned to implement the INSpIRA method as a proof of concept. This implementation should be used to execute and evaluate the INSpIRA method and workflow with at least two case studies from different industrial domains. Further the INSpIRA method and the obtained results should be compared with other methods and tools based on common characteristics, such as proposed in other surveys [4] [3]. The methods used to identify security risks could be evaluated based on the attack vectors described by the Information Security Forum (ISF) [5, 9] and by the Open Web Application Security Project (OWASP) [15]. Other properties of INSpIRA, like (a) the ability to test and (b) implement safety requirements of the SuD, (c) the detection of design failures or (d) the ability to recommend potential improvements, and (e) the Self-Evaluating GEE, could be evaluated using the proof of concept implementation.

However, it is planned to examine the following research objectives during the dissertation:

(1) Which tools and methods should be combined by the INSpIRA method to obtain a generic integrated security and risk assessment method?
(2) How could the toolchain be designed to efficiently support the cross-domain design, development, implementation and maintenance of CPSs?
(3) Which software architecture and design should be chosen to realize the GEE and the communication between the GEE and the SuD?
(4) How to generate test cases, self-tests and safety/security mechanisms?
(5) How to design and implement the Engineering Toolchain?

Figure 3 shows the planned timeline for completion. The items from the following listing correspond to the markers in Figure 3.

A) Definition of the first use case and execution of the INSpIRA method using pen and paper to identify potential design flaws before the implementation start.
B) Architecture, design and implementation of the *Test Execution Engine* in a generic manner. The implementation should be evaluated using the first use case.
C) Implementation (and adaption) of the *Test Execution Engine* to implement the *Safe & Secure Self-Adaptive System* . The implementation should be evaluated using the first use case.
D) Definition of the second use case and execution of the INSpIRA method, again using pen and paper.
E) Evaluation of the GEE using both use cases.
F) Architecture, design and partial automation of the design process (the Engineering Toolchain) to identify potential design flaws. The first and/or the second use case should be used for the evaluation.
G) Increase the degree of automation and try to execute the whole workflow using the implemented Engineering Toolchain with both use cases.
H) Re-evaluate the usability of the Engineering Toolchain and the INSpIRA method.
I) Finish the PhD Thesis.

6 CONCLUSION

The proposed INSpIRA method addresses several open research questions like

i. a comprehensive tool support for the integrated safety and security development of CPSs [3],
ii. the capturing of domain expert's opinion [3],
iii. the automatic analysis of risk models [3],
iv. the automatic generation of test cases [3],
v. the recommendation of potential improvements [3],
vi. the triggering of recovery strategies on undesired events [3]
vii. and the incorporation of runtime system information to make the risk assessment more effective [4].

If this question are answered during the doctoral research, INSpIRA provides a starting point to build comprehensive and sustainable methods and tools to efficiently design, develop, implement and maintain safe and secure systems that provide both, a strong safety and security argument.

REFERENCES

[1] Larry Apfelbaum and John Doyle. 1997. Model Based Testing. (1997). www.teradyne.com/sst
[2] Yean-Ru Chen, Sao-Jie Chen, Pao-Ann Hsiung, and I-Hsin Chou. 2014. Unified Security and Safety Risk Assessment - A Case Study on Nuclear Power Plant. (2014). DOI: http://dx.doi.org/10.1109/TSA.2014.13
[3] Yulia Cherdantseva, Pete Burnap, Andrew Blyth, Peter Eden, Kevin Jones, Hugh Soulsby, and Kristan Stoddart. 2016. A review of cyber security risk assessment methods for SCADA systems. *Computers & Security* 56 (feb 2016), 1–27. DOI: http://dx.doi.org/10.1016/J.COSE.2015.09.009
[4] Sabarathinam Chockalingam, Dina Hadžiosmanović, Wolter Pieters, André Andre Teixeira, and Pieter van Gelder. 2017. Integrated Safety and Security Risk Assessment Methods: A Survey of Key Characteristics and Applications. (oct 2017). DOI: http://dx.doi.org/10.1007/978-3-319-71368-7_5
[5] Information Security Forum. 2015. Threat Horizon 2017: Dangers accelerate. (2015). https://www.securityforum.org/research/threat-horizon-2017-dangers-accelerate/
[6] Heinz Gall. 2008. Functional safety IEC 61508 / IEC 61511 the impact to certification and the user. In *2008 IEEE/ACS International Conference on Computer Systems and Applications*. IEEE, Doha, Qatar, 1027–1031. DOI: http://dx.doi.org/10.1109/AICCSA.2008.4493673
[7] Edward H Glaessgen and David S Stargel. 2012. The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. (apr 2012). DOI: http://dx.doi.org/10.2514/6.2012-1818
[8] Johannes Iber, Tobias Rauter, Michael Krisper, and Christian Kreiner. 2017. The Potential of Self-Adaptive Software Systems in Industrial Control Systems. In *Communications in Computer and Information Science*. Springer, Cham, Ostrava, Czech Republic, 150–161. DOI: http://dx.doi.org/10.1007/978-3-319-64218-5_12
[9] Information Security Forum. 2017. Threat Horizon 2019: Disruption. Distortion. Deterioration. (2017). https://www.securityforum.org/research/threat-horizon-2on-deterioration/
[10] APIS IQ-Software. 2017. Funktionale Sicherheit. (2017). https://www.apis.de/
[11] Jeffrey O. J.O. Kephart and D.M. David M. Chess. 2003. The vision of autonomic computing. *Computer* 36, 1 (jan 2003), 41–50. DOI: http://dx.doi.org/10.1109/MC.2003.1160055
[12] Nancy. Leveson. 2012. *Engineering a safer world : systems thinking applied to safety*. The MIT Press.
[13] Georg Macher, Harald Sporer, Reinhard Berlach, Eric Armengaud, and Christian Kreiner. 2015. SAHARA: A Security-Aware Hazard and Risk Analysis Method. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015*. IEEE Conference Publications, New Jersey, 621–624. DOI: http://dx.doi.org/10.7873/DATE.2015.0622
[14] Igor Nai Fovino, Marcelo Masera, and Alessio De Cian. 2009. Integrating cyber attacks within fault trees. *Reliability Engineering and System Safety* 94, 9 (sep 2009), 1394–1402. DOI: http://dx.doi.org/10.1016/j.ress.2009.02.020
[15] OWASP. 2017. OWASP Top 10 -2017 The Ten Most Critical Web Application Security Risks. (2017). https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
[16] RISI. 2008. RISI - The Repository of Industrial Security Incidents: Schoolboy Hacks into Polish Tram System. (2008). http://www.risidata.com/Database/Detail/schoolboy_hacks_into_polish_tram_system
[17] Giedre Sabaliauskaite and Aditya P. Mathur. 2015. Aligning Cyber-Physical System Safety and Security. In *Complex Systems Design & Management Asia*. Springer International Publishing, Cham, 41–53. DOI: http://dx.doi.org/10.1007/978-3-319-12544-2_4
[18] Christoph Schmittner, Thomas Gruber, Peter Puschner, and Erwin Schoitsch. 2014. Security application of failure mode and effect analysis (FMEA). In *Computer Safety, Reliability, and Security*. Springer, 310–325. http://link.springer.com/chapter/10.1007/978-3-319-10506-2_21
[19] Christoph Schmittner, Zhendong Ma, Erwin Schoitsch, and Thomas Gruber. 2015. A Case Study of FMVEA and CHASSIS as Safety and Security Co-Analysis Method for Automotive Cyber-physical Systems. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security - CPSS '15*. -, Singapore, 69–80. DOI: http://dx.doi.org/10.1145/2732198.2732204
[20] Max Steiner and Peter Liggesmeyer. 2013. Combination of Safety and Security Analysis - Finding Security Problems That Threaten the Safety of a System. (2013). https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/4360
[21] Ralph Weissnegger, Christian Kreiner, Markus Pistauer, Kay Römer, and Christian Steger. 2017. SHARC - Simulation and Verification of Hierarchical Embedded Microelectronic Systems. *Procedia Computer Science* 109 (jan 2017), 392–399. DOI: http://dx.doi.org/10.1016/J.PROCS.2017.05.407