A Quantitative Approach for the Likelihood of Exploits of System Vulnerabilities

Siddhartha Verma(✉), Thomas Gruber(✉), Peter Puschner(✉), Christoph Schmittner(✉), and Schoitsch Erwin(✉)

Austrian Institute of Technology, Giefinggasse 4, 1210 Vienna, Austria
{Siddhartha.Verma,Thomas.Gruber,Christoph.Schmittner,
Erwin.Schoitsch}@ait.ac.at,
peter@vmars.tuwien.ac.at

Abstract. Modern systems' transition towards more connected, information and communication technologies (ICT) has increased the safety, capacity and reliability of systems such as transport systems (railways, automotive) and industrial systems but it has also exposed a big additional surface for cyber attackers which makes it necessary to take in consideration general IT security concerns. Cyber-physical systems need more effort to consider safety critical IT security concerns. The safety impact of security compromises is evaluated in a semiquantitative manner because it is a relatively new area so there is not enough real data available to analyse attack rates quantitatively and the attack-vulnerability scenario is constantly changing because of adversary intelligence. This paper proposes an approach for the quantification of vulnerabilities based on learning from data obtained by concrete pattern implementations in safety-critical systems. This will allow combined analysis of safety and security.

Keywords: Security patterns · Co-analysis · Colored petri nets
Security and dependability

1 Introduction

1.1 Safety and Reliability

System safety has been treated in a systematic manner by Functional Safety standards since the 1990s; the first edition of the generic Functional Safety standard IEC 61508 [34] was issued in 1998. Since then, many domain-specific standards have been elaborated by IEC, ISO and CENELEC, most of them based on the generic concepts of IEC 61508, which relies on process quality for coping with systematic faults and on probabilistic concepts for stochastic hardware faults.

A large amount of work has been done for analysing the reliability and availability of the system e.g. [3,8,9]. In [3] the railway operation is analysed based

B. Gallina et al. (Eds.): SAFECOMP 2018 Workshops, LNCS 11094, pp. 1–13, 2018.
https://doi.org/10.1007/978-3-319-99229-7_16

on the failure and maintenance data obtained from the Swedish infrastructure manager for the automatic train control signalling system. The paper uses a classical state-space Markov chains approach with stochastic transition rates for analysis. Redundancy technique is widely used for increasing the reliability of systems especially in safety-critical systems, which require high reliability and availability. In [8] a hybrid TMR (triple modular redundancy) fault tolerant structure is analysed. A hybrid redundant structure integrates voting scheme, fault detection and reconfiguration together. It uses a partial detection stochastic Petri net model for the system and a state transition Markov chain model for analysis.

There are many approaches for modeling and analysis/simulation of the reliability and availability of systems such as Markov Chains [1–3], eDSPNs (extended deterministic and stochastic Petri nets), colored Petri nets, or Fault-trees. Dependability patterns (e.g. [20]) also exist, which can provide solutions to recurring problems in the design of the system to control faults and their consequences by detection of errors, recovery from errors or by masking the error.

1.2 Security and Security Patterns

Security patterns are best practices to handle recurring security problems. Similar to safety, a large amount of work is available on security patterns e.g. [10–14] which includes a catalog of security patterns and describes the problem (about all the possible attacks) and the solution (how to implement the security strategies efficiently). The availability of a large number of patterns and the consideration of a further increase due to new patterns make it necessary to classify patterns to make the application of patterns easy along the life cycle of software. In [15,16] classification approaches for design patterns have been explained. In [17–19] classification approaches for security patterns have been described. In [4], a classification of security patterns based on application domains (Enterprise, Network, User, Software, Cryptographic) is presented.

1.3 Security and Safety Interdependence

In former years, Functional Safety standards did not address security-related safety-critical failures in detail because the assumption was that safety-critical systems are separated from externally reachable network areas like the internet, from where attackers could compromise it. In recent years, some standard updates mention at least security, however, without treating the related threats to safety in more detail.

For a few years now, awareness about security threats has been rising, also caused by reports about actual attacks. Currently, new editions of Functional Safety standards which include security concepts at least to some extent are being elaborated. Complementary to them, there are now a couple of domain-specific security standards in elaboration, like, as an example for the industrial control domain, IEC 62443 [30], which gives guidance how to analyze the system

for security flaws, how to rate it regarding the necessary security level, and which security controls are appropriate to implement.

The security attacks can affect the dependability of the system. For example consider an industrial drive system with a motor controller and communication networks (this raises a security concern). Figure 1. shows a sequence diagram of the system with communication among three software applications (client, server and motor control). The reliability of the system cannot be ensured unless the given level of security is not provided against security attacks. Therefore the reliability strategy of the system needs to be properly integrated with the security strategy. In [21,22] a hybrid pattern approach is represented which consists of security and dependability patterns which provide a way for addressing both dependability and security concerns. But the availability of documents containing catalogue of such patterns is not enough and not easily available.

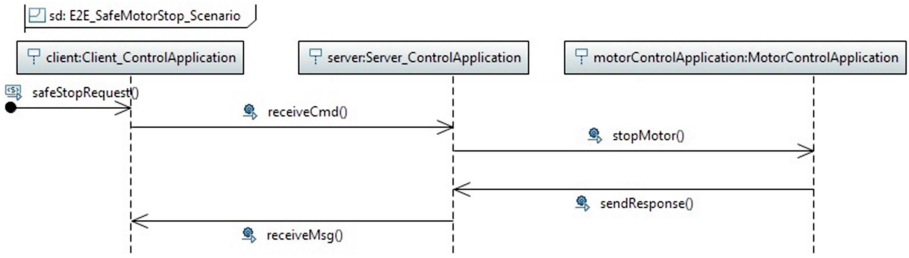

Fig. 1. Industrial drive sequence diagram

2 State of the Art Security and Dependability Co-engineering Approach

2.1 Safety Integrity Level and Tolerable Hazard Rate

According to Functional Safety standards, the maximum tolerable safety-critical failure rate of a component is related to the SIL (safety integrity level) assigned to it based on risk parameters (severity of the hazardous event, exposure time and controllability). The required SIL is determined in a HARA [24] (hazard analysis and risk assessment) in the concept phase of component development, and it is, according to functional safety standards like the generic IEC 61508 [34], directly assigned to a range for the rate of safety-critical failures λ as shown in Table 1.

Since security for information and communication systems in software based control systems in industrial, transport, health and other sector's is a relatively new area, there is not enough real data available to analyse attack rates quantitatively. Some works exist based on data-driven cybersecurity analytics to predict the attack rates, for instance [29] but only corresponding to a specific use case. Therefore, in the state of the art co-engineering approaches, the safety impact of security compromises has been evaluated in a semiquantitative manner, as shown in the next two sections.

Table 1. Maximum tolerable hazard rate (THR) ranges per SIL

SIL	λ [in h^{-1}]
4	$10^{-9} \leq \lambda < 10^{-8}$
3	$10^{-8} \leq \lambda < 10^{-7}$
2	$10^{-7} \leq \lambda < 10^{-6}$
1	$10^{-6} \leq \lambda < 10^{-5}$

2.2 SAHARA (Security Aware Hazard Analysis and Risk Assessment)

SAHARA (Security-Aware HARA) [6]) is an extension of HARA with STRIDE [25] (Spoofing identity, Tampering with data, Repudiation, Information disclosure, Denial of service and Elevation of privilege) to evaluate the impact of security issues on safety concepts. The goal of the STRIDE threat modelling approach is to analyze each system component for its susceptibility to threats and, subsequently, mitigate all threats to ensure system security. Similar to HARA, threats are quantified with reference to SIL analysis according to threat criticality (TC), resources (R) and know-how (K) that are required to pose the threat, to determine the security level (SecL) as shown in Table 2. The criticality of a security threat has 4 levels. Level 0, indicates that the security impact is irrelevant. Level 1 threats are limited to annoyances, such as reduction in service availability of non-safety-critical functions . Level 2 implies damage to products or manipulation of data or services and financial loss. Level 3 threats result in impacts on safety features (impact on human life). In comparison with HARA where the likelihood is expressed in terms of a failure rate, threat models determine likelihood semi-quantitatively in terms of required know-how and tools to launch an attack.

2.3 FMVEA (Failure Mode, Vulnerabilities and Effect Analysis)

FMVEA [1] is an extension of FMEA (Failure Mode and Effect Analysis [26, 27] to include threats and vulnerabilities in addition to accidental or stochastic failures. The system is divided into components, and failure/threat modes for each component are identified. For each failure/threat mode, the severity of the final effect and potential causes are examined. The severity of the effect is determined by the specific industry norms such as ISO 26262 [28]. In safety, the likelihood is expressed in terms of failure rate while, in security, the likelihood of an attack is expressed semi-quantitatively as an Attack Probability matrix (as shown in Table 3) which is determined from two factors: System susceptibility and Threat properties.

Motivation and Capabilities characterize the threat agent and their sum constitutes the Threat Properties. Values for Motivation are

Table 2. SecL Determination Matrix

R	K	TC			
		0	1	2	3
0	0	0	3	4	4
	1	0	2	3	4
	2	0	1	2	3
1	0	0	3	4	4
	1	0	2	3	4
	2	0	1	2	3
2	0	0	1	2	3
	1	0	0	1	2
	2	0	0	0	1
3	0	0	0	1	2
	1	0	0	0	1
	2	0	0	0	1

1 = opportunity target,
2 = mildly interested,
3 = main target,
and for Capabilities
1 = low,
2 = medium,
3 = high.

The sum of Reachability and Unusualness of the system characterize the System Susceptibility. Reachability is rated 1 = no network,
2 = private network, or
3 = public network,
and Unusualness
1 = restricted,
2 = commercially available, and
3 = standard.

Finally, the sum of System Susceptibility and Threat Properties yields the semi-quantitative value for the Attack Probability, as can be seen in Table 3.

3 Security and Dependability Co-engineering Approach - Proposed

In this section we proposed a co-engineering approach for analysing security and dependability, considering their interdependence. In Sect. 3.1, the Architectural Analysis for Security (AAFS) [3] approach has been illustrated for designing a secure system based on the architecture of the system under consideration. Section 3.2 mentions categories of component failures and explains why

Table 3. Estimation table for attack probability

System susceptibility	Threat-prop.				
	2	3	4	5	6
6	8	9	10	11	12
5	7	8	9	10	11
4	6	7	8	9	10
3	5	6	7	8	9
2	4	5	6	7	8

stochastic HW failures and cyber-attacks are considered for analysis. Section 3.3 discusses the idea of combining stochastic failure rates and attacks (successful exploitation of vulnerabilities) as a co-analysis approach. Section 3.4 discusses in detail a quantitative approach for the likelihood of exploits of vulnerabilities which makes it possible to combine the likelihood of (security-related) attacks with stochastic failure rates, allowing a quantitative safety-security co-analysis. The example presented in Sect. 3.1 will also be used to illustrate how the concrete implementation of security patterns can be used to generate enough data which will be used for learning how vulnerabilities can be quantified at discrete level based on vulnerability factors (system susceptibility and threat property as discussed in Sect. 2.3).

3.1 Architectural Security Analysis

Till recently most of the focus in security has been on secure coding to address the security concerns, which is evident by the presence of diverse static and dynamic code analysis tools. But by focussing only on coding it is not possible to achieve a high level of security. Architectural designs can sometimes overcome most of the complex coding issues very easily. Therefore, architecture has the central role in ensuring security.

We will use the AAFS approach for designing a secure system. AAFS proposes a combination of three approaches to design security. These approaches are Tactic-oriented Architectural Analysis (ToAA), Pattern-oriented Architectural Analysis (PoAA) and Vulnerability-oriented Architectural Analysis (VoAA).

ToAA - Tactic is the realization of design intention; it involves architects' experience and expertise about the system architecture instead of detailed analysis. Depending on the system architecture and requirements, the architect decides if the tactic is applied or not, and, if applied, whether there is any weakness or not. The tactic covers all possible design approaches available to the architect, so they can function as a design checklist.

For example a case study is presented in [3] about the Open Electronic Medical Record (OpenEMR) Project which demonstrates the use of AAFS, in ToAA phase with the architect's expertise it was realised that there is a weakness in the tactic 'verify message integrity'.

PoAA - Patterns are proven solutions to recurring problems. The pattern associated with a weak tactic realised in the ToAA phase is investigated and applied, if already applied evidence for system-wide adoption and proper implementation needs to be checked. Continuing the above example, the 'verify message integrity' tactic mentioned above is commonly associated with the 'intercepting validator' pattern which verifies inputs according to validation rules.

VoAA - Vulnerabilities are weaknesses in a software-intensive system that attackers exploit to breach security. Software vulnerabilities comprise coding errors and design flaws. Vulnerabilities are a measure of software security. Pattern implementation and coverage in PoAA can be checked during VoAA by exposing the architect to well-known software vulnerabilities (such as penetration testing) that can be addressed by the implemented pattern.

The list of known software vulnerabilities can be found at 'CWE (Common Weakness Enumeration) View- Architectural Concepts' [31]. For instance corresponding to the 'validate input' pattern, the list of possible vulnerabilities can be found at 'CWE- Architectural Concepts- Validate Inputs'. For example, some of the listed vulnerabilities associated to the 'validate input' strategy are 'improper neutralization of input during webpage generation (CWE-79)', 'improper neutralization of special elements used in SQL command (CWE-89)', and 'improper neutralization of special elements used in an OS Command (CWE-78)'.

Instances of the vulnerabilities can be found in the CVE list at National Vulnerability Database [33]; this list also evaluates the score of the vulnerability instance based on CVSS (Common Vulnerability Scoring System) which consists of vulnerability factors such as attack complexity, privileges required, user interaction and attack vector (which is the same as the susceptibility factor of FMVEA) .

From the list, the vulnerabilities relevant to the system under consideration are identified and, based on the vulnerability scores from FMVEA and CVSS, critical vulnerabilities are identified that need to be checked to provide the required reliability and availability for the system. Availability plays a role for on-demand safety functions, while reliability for continuous-mode safety functions.

3.2 How Components Can Fail

Components can fail (in particular also in a hazardous manner) due to the following causes:

1. safety-related failures, i.e.
   a. software errors (these correspond to systematic failures),
   b. systematic hardware failures (due to faulty construction or manufacturing), and
   c. stochastic hardware failures (due to ageing or wear-out), or
2. successful (security-related) attacks, which cause safety-relevant functions to fail.

All of these four categories, 1.a., 1.b., 1.c and 2., contribute to the overall failure rate. The Functional safety standards, which excluded security aspects until recently, needed to treat only the three above listed categories 1.a., 1.b. and 1.c., i.e. the safety-related faults only.

Regarding category 1.a. (software errors), the standards assume that the rate of failures caused by software faults is sufficiently reduced by Functional Safety standard-compliant development processes and, thus, they need not be further considered in the overall failure rate. For category 1.b. (systematic hardware faults), the assumption is that sufficient hardware testing reduces their contribution to the overall failure rate also to a negligible magnitude. (There are, however, also approaches treating software failures as a random failure rate. For our approach, we excluded them at the current stage from the considerations).

What remains is category 1.c., the stochastic hardware failure rate, which can be treated with statistical methods. This safety issue-related partial failure rate is what Functional Safety standards and also this paper are dealing with, and, in addition, we need to consider security-related failures caused by successful attacks against vulnerabilities. Those can be hardware failures (for example caused by overload like in the Stuxnet attack) or software failures (due to maliciously implanted malware code or tampered data).

3.3 How Can We Bring Together Stochastic Failure Rate and Attacks

As discussed in Sect. 2.1 the severity of impact on safety is one of the main factors for deciding on the SIL, which is directly assigned a maximum tolerable failure rate. So in order to meet a required SIL, the combined failure rate due to security attacks and hardware failures must be less than the maximum tolerable hazardous failure rate. But as discussed in Sects. 2.2 and 2.3 the existing co-engineering approaches use semi-quantitative values for attack probability instead of attack rates, therefore a decision based on experience and expertise is made to decide if the likelihood of critical failures due to attacks is less than the maximum tolerable failure rate. If not, the vulnerability exploited by the attack must be checked by a proper security strategy, in order to ensure the required safety level. The goal of the approach presented in this paper is to get a mapping between

1. the semi-quantitative values for threat probabilities of threat modes of components (in the architecture), which are obtained by applying the FMVEA, and
2. the quantitative tolerable hazardous failure rate values of the components associated to their SIL. This will allow combining the safety and the security analyses of the of the system with a quantitative risk assessment scheme that makes the partial hazard rates related to safety issues and to security-related attacks comparable. The critical vulnerabilities need to be checked to ensure safety requirement of the system is met.

3.4 Approach for Quantification of Threats Related to Vulnerabilities

To be precise by quantification of threats, we can calculate the combined failure rate as sum $\lambda_c = \lambda_{HW} + \lambda_{Attack}$ only under the assumption that there are no common cause failures between stochastic HW failures and cybersecurity attack-induced failures. This assumption seems reasonable for most cases. As for the Stuxnet attack, however, there is some inter-relation by the "normal" λ of the centrifuges and the damage caused by overstress. We consciously neglect this imprecision for now as is seems to apply only for rare cases.

A large collection of security patterns is present as discussed in Sect. 1.2, which addresses several vulnerabilities. If we take into consideration a given concrete implementation of a security pattern for a component in the system architecture (which has a required SIL) then we can assume that those potential security-induced failure modes against which the component is protected by the security pattern will not occur. In this sense, we assume that threats to vulnerabilities which are protected by a given security pattern are critical to the system (which has a certain SIL requirement).

As discussed in the FMVEA, the idea is to analyse the system and provide the required safety integrity level (SIL) based on this analysis by implementing the required safety strategy and checking (implementing security pattern to secure) these vulnerabilities for the exploit, we can deduce that these (neutralized) vulnerabilities are critical for the corresponding SIL. E.g. let's say the system requires SIL 2, so the combined tolerable hazardous failure rate is $10^{-7} \leq \lambda < 10^{-6}\ h^{-1}$. As an example, let's assume that the stochastic failure rate due to hardware faults analysed is sufficiently low (low means as in comparison to the system safety requirement of SIL 2, which requires a tolerable failure rate $10^{-7} \leq \lambda < 10^{-6}\ h^{-1}$ or less), e.g. $10^{-9} \leq \lambda < 10^{-8}\ h^{-1}$, then the security strategy must provide a reliability $\geq 10^6$ h.

We can understand the concept in the above two paragraphs by the OpenEMR Project example as discussed in Sect. 3.1. The openEMR project on analysis found a need for implementation of a validator security pattern for following four vulnerabilities, which the experts found to be of highest priority. And, the security strategy applied initially does not seem to cover these vulnerabilities or it cover them only partially.

Four vulnerabilities of highest priority were:

1. Improper neutralization of special elements used in a SQL command
2. Improper neutralization of input during webpage generation
3. Improper neutralization of script-related HTML tags in a web page
4. Improper neutralization of alternate XSS syntax

The mentioned four vulnerabilities can be scored based on vulnerability factors (susceptibility and threat property as discussed in Sect. 2.3 FMVEA) based on the known potential threats that can exploit these vulnerabilities. Let's assume the required SIL for openEMR system is '2'. Now since these vulnerabilities are critical for system which has required SIL '2' because they have

been secured by pattern. We will assume that the tolerable failure rate due to these vulnerabilities are higher than $10^{-6}\ h^{-1}$.

Similarly other security patterns have implementations in different systems which have different SIL requirements and different system conditions (such as system reachability and unusualness), this will allow us to generate a lot of data (tolerable failure rate and vulnerability factor's score). And this data will we used to learn how to map the vulnerability factor (susceptibility and threat property) scores to failure rates due to vulnerability exploits. After the learning algorithm will be realized we will be able to predict in which SIL zone the exploit rate of the vulnerability ranges as shown in Fig. 2. E.g. if it ranges in SIL 3, we will use $10^{-7}\ h^{-1}$ as the exploit rate, considering worst case scenario ($10^{-7} > 10^{-8}$).

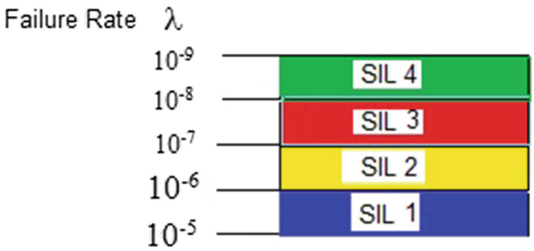

Fig. 2. Discrete Quantification of Vulnerabilities Based on Learning

The quantification will be at discrete level corresponding to the maximum tolerable hazardous failure rate (assigned as SIL requirement).

So the final outcome of this analysis would be, that, based on vulnerability score factors, the range of the rate of exploits can be predicted. E.g. we will be able to predict that, if the given vulnerability (based on its vulnerability factor's score) exploit range is SIL 3 or SIL2. If it is SIL 2, this means that the exploit rate will be greater than $10^{-7}\ h^{-1}$ and less than $10^{-6}\ h^{-1}$. But considering the worst case scenario for the co-analysis, we will take the attack rate $= 10^{-6}\ h^{-1}$.

It is important to note and realise that the rate of exploits of vulnerabilities obtained by the above mentioned way is not absolute. All data generated by the concrete implementation of security patterns in safety critical systems, which will be used for the quantification of the likelihood of vulnerability exploits (by a learning algorithm), is based on the expert decision / recommendation that certain vulnerabilities (based on vulnerability factors) are critical for the given system with a certain SIL requirement. This approach will allow us to use all such already established expert decisions in a comprehensive and homogenous way for the quantification of vulnerabilities.

Such a quantification of vulnerability as an exploit rate similar to the failure rate will allow us to consider safety and security under the same umbrella and categorize critical vulnerabilities that need to be checked to ensure the required safety level. The quantitative approaches mentioned in Sect. 1.1 would be possible to use. This will provide more certainty and reliability to our analysis

as compared to a semi-quantitative approach for considering attack occurrence (rate), taking into consideration the real existing use cases of the security pattern implementations as a base. The learning algorithm is still to be realised.

The approach described in this paper is still on a conceptual level in that the learning algorithm is still to be realised.

3.5 Data from Security Patterns Implementations

A large collection of security patterns is present as discussed in Sect. 1.2, which addresses several vulnerabilities. If we take into consideration concrete implementation of these security patterns in safety-critical systems (with a required SIL) then we can assume that the specific vulnerabilities checked by these patterns are critical to the system, and that the failure rate due to the exploitation of these vulnerabilities is higher than the one required as discussed in detail in Sect. 3.4. Thus, considering the implementation of patterns in safety-critical systems will provide us with enough data to do a data-driven analysis for the quantification of vulnerabilities based on the score of vulnerability factors.

4 Conclusion and Further Work

In this paper an approach for developing a quantitative metric for the likelihood of exploits of system vulnerabilities has been drafted. The next step will be running a learning algorithm on use cases with concrete implementations of patterns for safety critical systems in order to obtain the quantitative values. After this, the idea is to use architecture based security design as mentioned in detail in Sect. 3.1, in order to to design a secure and safe system. During the VoAA the vulnerabilities will be identified and quantified based on the vulnerability factors as discussed earlier. Then idea is to combine both failure/maintenance and attack/recovery using stochastic colored petri nets where the system's reliability and availability can be analysed or simulated e.g. using tools like TimeNET. A way of modelling and quantitative evaluation of vulnerabilities based on stochastic petri nets is mentioned in [35]. A way of modelling failures/maintenances for reliability and safety evaluation using petri nets is shown in [36]. As a result the critical vulnerabilities can be marked and checked using patterns for providing the required availability and reliability.

Yet another benefit can be obtained from having quantitatively comparable partial hazard rates resulting from stochastic hardware failures and failures induced by cybersecurity attacks: It supports strongly safety-security co-engineering, which strives to achieve a balanced set of measures aiming at security improvements and safety-oriented mitigation measures, which are often contradictory. They can be selected in an optimal way based on a trade-off analysis supported by a quantitative metric.

Acknowledgements. The work published here is based on research in the AMASS project that has been funded by the ECSEL Joint Undertaking under Grant Agreement number 692474.

References

1. Schmittner, C., Gruber, T., Puschner, P., Schoitsch, E.: Security application of failure mode and effect analysis (FMEA). In: Bondavalli, A., Di Giandomenico, F. (eds.) SAFECOMP 2014. LNCS, vol. 8666, pp. 310–325. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10506-2_21
2. Whitt, W.: Continuous Time Markov Chains, Department of Industrial Engineering and Operations Research, Columbia University (2013)
3. Morant, A., Gustafson, A., Söderholm, P., Larsson-Kråik, P.O., Kumar, U.: Safety and availability evaluation of railway operation based on the state of signaling systems. Proc. Inst. Mech. Eng. Part F J. Rail Rapid Transit **231**, 226–238 (2017)
4. Ryoo, J., Kazman, R., Anand, P.: Architectural analysis for security. IEEE Secur. Priv. **13**(6), 52–59 (2015)
5. Bunke, M., Koschke, R., Sohr, K.: Organizing security patterns related to security and pattern recognition requirements. Int. J. Adv. Secur. **5**, 46–67 (2012)
6. Satty, T.L.: The Analytical Hierarchy and Analytical Network Measurement Process: Applications to Decisions under Risk (2008)
7. Macher, G., Sporer, H., Armengaud, E., Kreiner, C.: SAHARA: A Security-Aware Hazard and Risk Analysis Method (2015)
8. Liu, Z., Liu, Y., Cai, B., Liu, X., Li, J., Tian, X., Ji, R.: RAMS Analysis of Hybrid Redundancy System of Subsea Blowout Preventer Based on Stochastic Petri Nets (2013)
9. Mustafiz, S., Sun, X., Kienzle, J., Vangheluwe, H.: Model-driven assessment of system dependability. Softw. Syst. Model. **7**, 487–502 (2008)
10. Schumacher, M., Fernandez, E.B., Hybertson, D., Buscmann, F., Sommerlad, P.: Security Patterns : Integrating Security and System Engineering. Software Design Patterns. Wiley, Hoboken (2006)
11. Haldikis, S., Chatzigeorigou, A., Stephanides, G.: A practical evaluation of security patterns (2006)
12. Steel, C., Nagappan, R., Lai, R.: Core Security Patterns : Best Practices and strategies for J2EE : Web Services and Identity Management (2014)
13. Kienzle, D.M., Elder, M.C., Tyree, D., Edwards-Hewitt, J.: Security Patterns repository version 1.0 (2002)
14. Dougherty, C., Sayre, K., Seacord, R.C., Svoboda, D., Togashi, K.: Security Design Patterns (2009)
15. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: Pattern-Oriented Software Architecture: A System of Patterns. Wiley, New York (1996)
16. Shi, N., Olsson, R.A.: Reverse engineering of design patterns from Java source code (2006)
17. Konrad, S., Cheng, B.H., Campbell, L.A., Wasserman, R.: Using security patterns to model and analyse security requirements (2003)
18. Rosado, D.G., Gutierrez, C., Fernandez-Medina, E., Piattini, M.: Security patterns related to security requirements (2006)
19. Washizaki, H., Fernandez, E.B., Maruyama, K., Kubo, A., Yoshioka, N.: Improving the classification of security patterns (2009)
20. Saridakis, T.: A system of patterns for fault tolerance. In: Proceedings of the EuroPLoP Conference (2002)
21. Buckley, I.A., Fernandez, E.B., Larrondo-Petrie, M.M.: Patterns combining reliability and security (2011)

22. Hamid, B.: Modelling of secure and dependable applications based on a repository of patterns : The SEMCO approach
23. Charlwood, M., Turner, S., Worsell, N.: A methodology for the assignment of SILs to safety-related control functions implemented by safety-related electrical, electronic and programmable electronic control system of machines : prepared by Innovation Electronics UK Ltd and Health and Safety Laboratory HSL (2004)
24. Stolte, T., Bagschik, G., Reschka, A., Maurer, M.: Hazard Analysis and Risk Assessment for an Automated Unmanned Protective Vehicle (2017)
25. Microsoft Corporation : The stride threat model (2005)
26. Reifer, D.J.: Software Failure Modes and Effects Analysis (1979)
27. Haapanen, P., Helminen, A.: Failure mode and effect analysis of software-based automation systems (2002)
28. ISO - International Organization for Standarization: ISO 26262 Road vehicles Functional Safety (2011)
29. Zhan, Z., Xu, M., Xu, S.: Predicting cyber attack rates with extreme values (2015)
30. IEC 62443 : Industrial communication networks - Network and system security (2010)
31. cwe.mitre.org: Common weakness enumeration view : Architectural Concepts (2018)
32. TimeNET : A software tool for the performability evaluation with stochastic and colored petri nets. https://timenet.tu-ilmenau.de/template/index
33. Mell, P., Scarfone, K., Romanosky, S.: A Complete Guide to the Common Vulnerability Scoring System (2007)
34. IEC- International Standards and Conformity Assessment for all electrical, electronic and related technologies. http://www.iec.ch/functionalsafety/standards/page3.htm
35. Flammini, F., Marrone, S., Valeria, V.: Petri Net Modelling of Physical Vulnerability (2013)
36. Pinna, B., Babykina, G., Brinzei, N., Petin, J-.F.: Using Coloured Petri Nets for integrated reliability and safety evaluations (2013)