Security-Oriented Fault-Tolerance in Systems Engineering: A Conceptual Threat Modelling Approach for Cyber-Physical Production Systems

Iris Gräßler[1](✉), Eric Bodden[2], Jens Pottebaum[1], Johannes Geismann[2], and Daniel Roesmann[1]

[1] Heinz Nixdorf Institute – Product Creation, Paderborn University, Fürstenallee 11, 33102 Paderborn, Germany
{iris.graessler,jens.pottebaum,daniel.roesmann}@hni.upb.de
[2] Heinz Nixdorf Institute – Secure Software Engineering, Paderborn University, Fürstenallee 11, 33102 Paderborn, Germany
{eric.bodden,johannes.geismann}@hni.upb.de

Abstract. Faults in the realization and usage of cyber-physical systems can cause significant security issues. Attackers might exploit vulnerabilities in the physical configurations, control systems, or accessibility through internet connections. For CPS, two challenges are combined: Firstly, discipline-specific security measures should be applied. Secondly, new measures have to be created to cover interdisciplinary impacts. For instance, faulty software configurations in cyber-physical production systems (CPPS) might allow attackers to manipulate the correct control of production processes impacting the quality of end products. From liability and publicity perspective, a worst-case scenario is that such a corrupted product is delivered to a customer. In this context, security-oriented fault-tolerance in Systems Engineering (SE) requires measures to evaluate interdisciplinary system designs with regard to potential scenarios of attacks. The paper at hand contributes a conceptual threat modelling approach to cover potential attack scenarios. The approach can be used to derive both system-level and discipline-specific security solutions. As an application case, issues are focused on which attackers intend to exploit vulnerabilities in a CPPS. The goal is to support systems engineers in verification and validation tasks regarding security-oriented fault-tolerance.

Keywords: Systems engineering · Threat modelling · Scenario-based analysis

1 Motivation and Approach

Faults in the realization and usage of cyber-physical systems (CPS) can cause significant security issues. Fault tolerance is the ability of a system to tolerate faults without limitations in its performance of functions [1, 2]. The term ‘fault’ refers to unpermitted deviations of characteristics of a technical system. Thus, a fault represents a state of the system; fault tolerance requires fault detection, fault diagnosis and fault management by

A. Bartoszewicz et al. (Eds.): PCC 2020, AISC 1196, pp. 1458–1469, 2020.
https://doi.org/10.1007/978-3-030-50936-1_121

design [2]. Typical perspectives in design phases are focused on (potential) impacts of faults in terms of reliability, availability and safety [2].

At the same time, designers target security of a technical system. The system should withstand intentional attacks [3]. Assuming 'security' as a system characteristic, a fault-tolerant system needs to perform as usual even in the event of such an attack, without causing any failures or malfunctions. Attackers might exploit vulnerabilities in the physical configurations, control systems, or accessibility through internet connections. For CPS, two challenges arise: Firstly, discipline-specific security measures should be applied. Secondly, new measures have to be created to cover interdisciplinary impacts.

Cyber-Physical Production Systems (CPPS) [4] are treated as a specific field of research. Two levels have to be considered, the production process performed by the system and the product produced within that process. Faults might impact both levels. For instance, faulty software configurations in CPPS might allow attackers to manipulate the correct control of production processes, impacting the quality of end products. From liability and publicity perspective, a worst-case scenario is that such an attack is not detectable as a process deviation and, consequently, a corrupted product is delivered to a customer. Examples are already given in various branches [5].

In this context, security-oriented fault-tolerance in Systems Engineering (SE) requires measures to evaluate interdisciplinary system designs with regard to potential attack scenarios. Requirements of systemic threat modelling are documented in Sect. 3. An analysis of related work indicates that current methods and tools are specialized to single domains even when they are motivated by interdisciplinary fault analyses (Sect. 4). This paper contributes a conceptual threat modelling approach to cover potential attack scenarios from a holistic system's application perspective (Sect. 5). The approach can be used to derive both system-level and discipline-specific security solutions. As an application case, issues are focused on which attackers intend to exploit vulnerabilities in a CPPS. The goal is to support systems engineers in verification and validation tasks regarding security-oriented fault-tolerance.

2 State of the Art

This paper focuses on challenges in the design of Cyber-Physical Production Systems (CPPS) as a special type of integrated Cyber-Physical systems (CPS). The approach is driven by two main demands: enabling traceable fault-tolerance from system to component level and, at the same time, ensuring security-oriented fault-tolerance from component to system level. Domain and demands are concretized in the following sections.

2.1 CPS and CPPS

CPS are highly networked technical objects that contain embedded systems, can exchange digital information and can use other services. With appropriate sensors, they are linked with the environment, save available data, evaluate them with the help of services and influence the physical world with actuators (cf. Figure 1). They are connected by means of internet technologies [4].

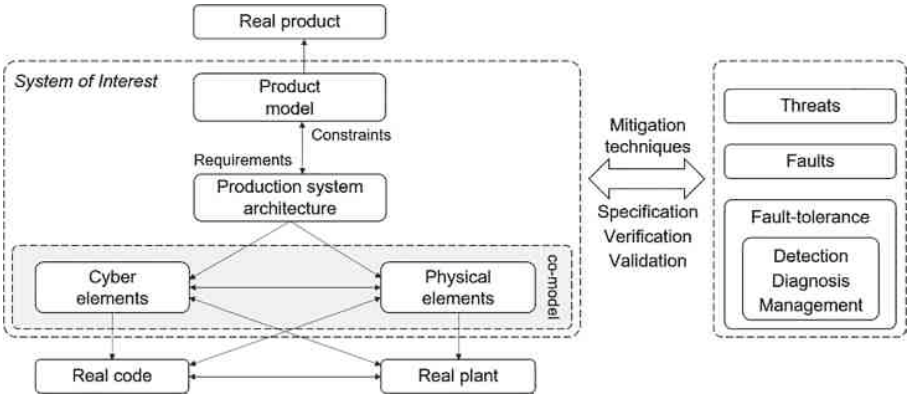

Fig. 1. Terminology and relationships of CPPS [4] and fault-tolerance [2]

Lee defines the term as follows: "Cyber-Physical Systems (CPS) is an integration of computation with physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa." [6] The term CPPS is derived from the definition of CPS. Accordingly, a CPPS is a production system that has the characteristics of a CPS. Production systems are socio-technical systems that transform input into output through value-adding and supporting processes. For this purpose, resources like machines, transport systems etc. are used. The objective of running a production system is the capability to produce the right products at the right time in the defined quality at reasonable costs. In contrast to conventional production systems, CPPS are communicate via internet. They are able to collect process information, which allows an integrated cognition and artificial intelligence [7]. The complete implementation of a CPPS may be entitled "Smart Factory". Materials, products and systems are equipped with sensors and actuators and ideally organized without human intervention [8].

Security in CPPS is mainly focused on issues related to information and communication technologies, especially smart grid. This is confirmed by Nguyen et al. who conducted a systematic study on approaches of Model Based Security Engineering [9]. Sadeghi et al. emphasize that security issues have to be addressed on all CPPS abstraction layers [5]. They elaborate on electronics, software and even humans. In addition to their approach, machines with their physical dimension should be included within CPPS system boundaries.

2.2 Security and Fault-Tolerance in Systems Engineering

Fault-tolerance implies detection, analysis and management of faults. These functionalities have to be integrated across all system-levels. Systemic fault-tolerance is mainly focused from the perspectives of safety, requiring system integrity. Typically, methods like Failure Modes and Effects (and Criticality) Analysis (FMEA/FMECA), Hazard Analysis as well as Event and Fault Tree Analysis (ETA/FTA) are combined for system safety and reliability in design phase [2]. In late 1990's, the Systems Security Engineering (SSE) project team identified capability levels and relevant process areas of SSE

without specifying particular methods or processes [3]. As a result, SSE related practices were captured in ISO/IEC 21827. The model focuses specifically on information systems security. For the paper at hand, considerations of security risks, threats (including, for instance, threat agent's capabilities) and system security verification and validation (V&V) are relevant foundations.

Searching the library of the Design Society shows only very few publications related to fault-tolerance. Deyter et al. provide a case study of applying FTA on the principle solution using application scenarios in an early design phase of mechatronic systems [10]. Kolberg et al. propose a methodology to support design changes; they build up on six established "methods" from problem formulation to fault-tolerance [11]. Extending the literature review to google scholar, applications in specific branches can be added. Rostami et al. present brief insights into scenarios of threats propagating along globalized semiconductor supply chain [12]. Isaksson and Ritchey target system-level threats in their domain of nuclear facilities [13] where artefacts like the Design Basis Threat (DBT) are obligations.

2.3 Security-Oriented Fault-Tolerance

Security-orientation means coverage of various properties according to established security taxonomies (cf. [14]). The library of the Design Society does not contain any paper referring to both 'faults' or 'malfunctions' and 'security'.

In general, the reason for a fault is in the first place not important to the system designer. Hence, the term "fault" is used to cover all kind of faults. In contrast to safety, security-oriented faults describing faults caused by an (successful) attack to the system and are always based on a malicious intention. Since fault-tolerance describes the degree a system can handle faults without limitations to its behaviour, we argue that it is conceptual related to the term "threat modelling" [15] of the security domain. In threat modelling, potential threats to the system but also mitigations to these threats are defined. To determine which threats are relevant, security risk assessment is an essential part of a threat modelling approach. However, the only way to remove a threat completely from the system is to remove all affected system parts. Hence, when addressing potential threats, these threats are only mitigated and, therefore, a certain probability for these threats remain. Since it is not possible to prevent attacks, the goal of threat modelling is to find design decisions decreasing the probability of a successful attack and, therefore, increasing the systems security-oriented fault tolerance.

Moreover, following the concept of "defence-in-depth", a systematic threat modelling approach takes scenarios into account in which countermeasures are circumvented by an attacker or simply not correctly implemented. One essential part is to split the system into so-called "trust-zones" and to restrict privileges for each zone or system part to the minimum ("Principle of Least Privilege"). This technique helps to reduce the impact of an attack and, therefore, make the system more tolerant to certain attacks.

Hence, threat modelling can be used to systematically determine the degree a system is protected against threats or attacks respectively. In combination with a methodology ensuring continuous threat modelling along the whole engineering process, it is the root for security-oriented fault-tolerance.

In addition, when securing a system, it is always assumed that not all attacks can be known beforehand (for instance, if a used cryptographic library gets outdated). Thus, for some scenarios countermeasures are implemented that focus on detecting (and reporting) attacks instead of prevent them. Prominent examples are here Intrusion Detection Systems (IDS) or cryptographic signatures (e.g. to ensure the integrity of assets). Hence, threat modelling also covers threat scenarios that may occur in the future.

3 Requirements of Systemic Threat Modelling

When designing a method for security-oriented fault tolerance in the context of highly interdisciplinary development, there are several requirements such a method should take into account. They are derived from the different perspectives analyzed in Sect. 2. Firstly, there are requirements regarding the methodology itself focusing on the process as well as the context in which the method is applied. The method should

(R1) support all stages/steps of the engineering process, for instance, INCOSE Systems Engineering processes [16], VDI 2206V model of product and respective production system development (current state [17] and evolution [18]) up to operation and decommissioning stages,
(R2) cover informal and formal security requirements and threat descriptions, and
(R3) support interdisciplinary threat elicitation, risk analysis and mitigation.

Furthermore, it is essential that a method is not built from scratch but can be integrated into existing development frameworks. Therefore, the following requirements are important regarding the contextualization of the method. The method should

(R4) utilize standard vocabularies across disciplines, branches and system-levels,
(R5) be based on standard modelling methods/languages, and
(R6) take into account standard branch-specific scenarios documented in guidelines, directives or even norms.

In addition, a threat model can provide a backbone when collecting, persisting, and analyzing potential threats. The threat model should

(R7) cover both product security and CPPS security and their dependencies,
(R8) be extendable for new scenarios,
(R9) target all levels of system architectures bi-directionally (breaking down scenarios from system to component level, aggregating security assessments from component to system level),
(R10) allow for traceability of issues from faults to threats to misuse cases, and
(R11) be formalized to support derivation of Verification & Validation/test specifications.

4 Related Work

Since decades, scenario analysis is applied to inform decision making in a comprehensive and tangible way [19], including approaches supporting systems engineers [20]. Threat

scenarios can be approached from three different perspectives: prognosis of potential future threats in general, modifications of an application scenario in terms of possible or even probable stories [21] as well as modifications of a system specification that describes a system state in case of an attack [22]. The library of Design Society does not contain any publication which covers both "threat" and "model*" as keywords. Three search results are only focused on threats regarding disruptive innovation, value-centric development and IP protection. Extending the review of related work on scenario based interdisciplinary approaches, google scholar indicates only partial overlaps. Kim and Cha apply a scenario based threat modelling approach on a broadband convergence Network [23] specifically focusing on information systems. Detecting insider threats is a special field of applications (cf., for instance, [24]).

SysML-Sec [25] is a model-driven engineering approach that "aimed at fostering the collaboration between system designers and security experts" [26] during all phases of the development of cyber-physical systems. It is based on SysML and provides customized SysML diagrams to describe security-related system parts. It also provides a methodology for a systematic development focusing on closing the gap between safety and security modelling. It covers steps for (security) requirements engineering, system design, design validation, and also (partially) code generation for the target system. In contrast to our approach, SysML-Sec focuses only on the development of CPS but does not take the special requirements of CPPS into account nor it takes the security requirements for the product into account.

5 Scenario-Based Security-Oriented Fault-Tolerance Validation

In this section, the proposed approach is presented based on the development steps shown in Fig. 2 and its application in a laboratory environment. The conceptual approach combines scenario-based and security-oriented development steps and iterative Model Based Systems Engineering methodologies. Threat modelling techniques are integrated into existing MBSE methods targeting, on the one hand, requirements **R4** and **R5** but also **R1** since the threat modelling is refined along the whole development cycle.

5.1 Conceptual Approach

Assuming a system specification is available after development iteration (S_i), its fault-tolerance shall be validated regarding identified threats [22]. S_i is specified by SysML referring to discipline specific co-models (like UML for software and STEP for mechanics) according to [4].

System-level security requirements are based on **security policies** and **global security requirements**. Global means system independent, often companywide specifications. Since these requirements are adopted specifically for product and production system in early planning steps, it is essential that they can be defined informally first and are getting refined at later development steps (cf. **R2**). These might differ in terms of branches, markets, countries and other aspects, for instance based on data privacy regulations like the GDPR. We assume that the security policies and security requirements are elicited using existing methods based on the combination of system assets and threat

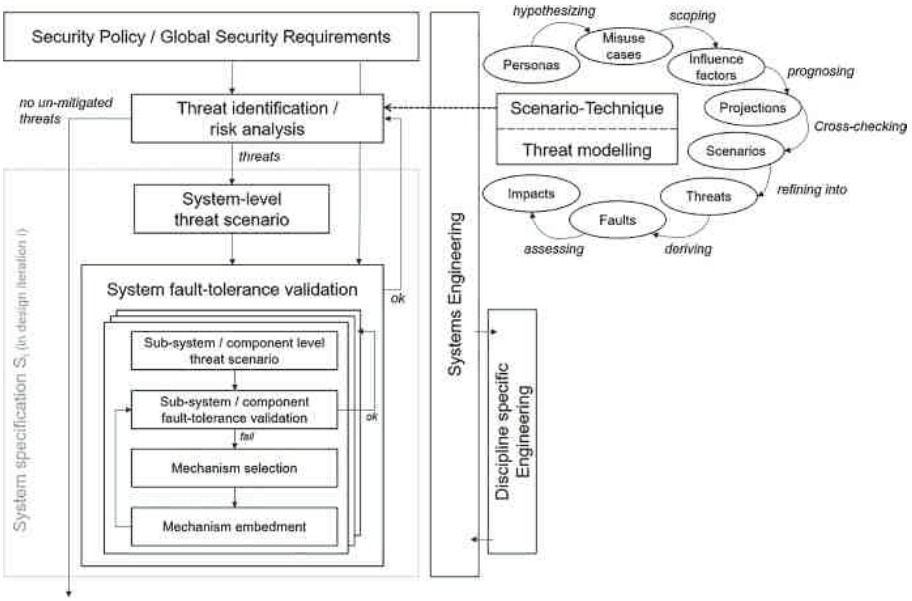

Fig. 2. Conceptual approach of fault-tolerance validation: Complementing Systems Engineering and discipline specific engineering by means of Scenario-Technique and threat modelling

categories like STRIDE [15] or common security properties like CIA [27] (cf. **R6**). On the one hand, they are complemented by discipline-specific security requirements. On the other hand, threat identification as an activity of risk analysis is performed (main path in Fig. 2). To ensure security on all levels, addressing a global security requirement has to be accomplished by different disciplines in combination (cf. **R3**).

The **Scenario Technique** is proposed as a method to estimate potential threats: Early phases of the scenario technique target artefacts like personas and abuse/misuse cases [28], future evolutions are estimated by influence factors and corresponding projections of possible futures which are aggregated into scenarios [29]. Influence factors are derived based on morphological analyses (cf. [13]) considering system-level security taxonomies (cf. [14]). **Scenarios** are used as descriptive models which are comprehensive for all stakeholders [19]. Scenarios are analyzed with regard to threats on system and sub-system/component levels. Resulting threats are formalized for uptake in engineering tools. Effects in terms of faults within the system/sub-system/component as well as impacts between systems (esp. production faults impacting products) can be derived.

A **system-level threat scenario** means a modification of S_i representing a threat causing a faulty state of the system. Once an initial set of threat scenarios are identified, these scenarios have to be refined to discipline-specific threats (describing which part of the scenario can be tackled by which discipline). Existing threat modelling techniques are integrated for each development stage (cf. [30]). Similar to [25], we suggest to use attack-defense graphs for documenting the refinement of threat scenarios into sub-scenarios and to define potential mitigation. This technique allows defining relations between the impacts on different system parts and different disciplines (cf. **R9**). Explicit modelling

of relationships allows traceability along all analysis stages. This applies especially to traceability of threats and impacts between production system and materials/products handled within corresponding processes and transferred into customer use cases later. While a threat might refer to an element of the production system (e.g., a manufacturing cell in a company's factory), the impact might become active in product use at a customer's location (cf. Sect. 5.2). For this reason, we argue that it is essential to apply a holistic threat model covering both the production system and the product in one threat model. When eliciting or refining threats and mitigations, the security experts have to tag involved disciplines but also which part of the system or the product is affected by a threat or responsible for the correct implementation of a mitigation (cf. **R7**). Noteworthy is that security engineers of one discipline may define other disciplines as responsible for a mitigation during the **discipline-specific Engineering.**

Since it is unsuitable to review the whole threat model, it is important to enable discipline-specific views on the model showing only the relevant parts and impacts to the engineer [31]. Creating new threats or mitigations for another discipline will open up new threat model parts that have to be discussed at the beginning of the next iteration (S_{i+1}) by all involved disciplines before it becomes approved for the threat model. Correspondingly, at each iteration step, it can be determined if all desired mitigations are (designed to be) in place. If a specific threat is not mitigated sufficiently, it may be refined again or delegated to other disciplines as well in the next iteration. When all threat scenarios are sufficiently addressed (which is when the fault-tolerance is high enough), the threat is marked as mitigated and the iteration stops. If there are no un-mitigated threats, the threat modelling stops until changes to the threat model or the system models are made and a new iteration is necessary. Using iterations that are update whenever a model changes helps keeping the threat model up to date and to integrate new threats and mitigations when needed (cf. **R8**).

5.2 Application Case in Cyber-Physical Production Systems

An application case is simulated in a laboratory environment at Heinz Nixdorf Institute in Paderborn. Within a realistic factory environment, exemplary mechatronic products are produced like remote controlled cars (RC cars) and small drilling machines [7]. The lab is established as a complete production system with manufacturing cells, assembly station/line, material flow sub-system and software. All manufacturing cells include different manufacturing techniques (milling, turning and 3D printing) and industrial robots (portal robot, articulated robot and cobot). The software architecture spans from product configuration and Enterprise Resource Planning (ERP) to manufacturing execution and process control. For the analysis of security-oriented faults and corresponding fault-tolerance measures, a scenario is chosen which covers intentional security breaches in a production system producing RC cars, threats that are implied for products and impacts for end customers. Even though the example is far from real cars, correspondences between the lab case and real applications can be analyzed.

A sample scenario is defined as follows (cf. Fig. 3): Amongst others, a persona is defined regarding the intention to reduce trust in products of automotive industries by provoking technical issues up to accidents of private cars. A sample misuse case

describes data manipulation causing the 3D printer to produce lower quality parts. Technically speaking, load-bearing capacity of parts is reduced by manipulating manufacturing parameters; standard non-destructive testing for quality assurance would not be capable of recognizing this threat before shipment to customers.

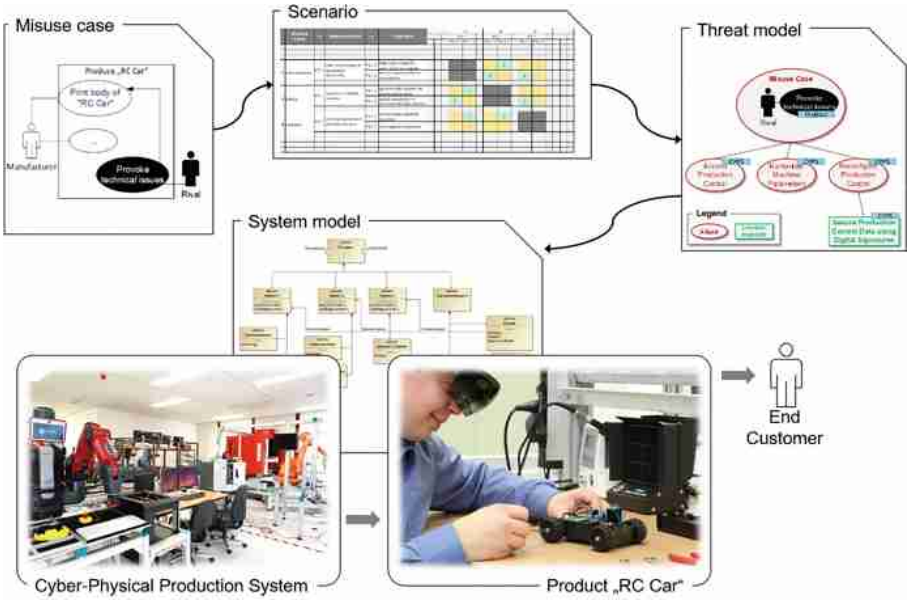

Fig. 3. Artefacts of scenario-based and security-oriented fault-tolerance validation

Scenario Technique is applied based on the assumption that relevancy, probability and business impact of misuse cases are influenced by various factors. Considering use cases as contextual frame for product usage, influence factors are collected in influence fields. Such fields subsume generic aspects like 'trust in technology in general' as well as specific one for automotive. In this sample scenario, 'interrelations with international politics' and 'technical expertise of potential attackers' were identified as relevant fields based on published studies in this domain. Cross-impact analysis results in clusters of active and/or passive factors. Projections are derived from statistics, simulations and expert workshops with emphasis on active factors. For instance, governments might tend to partnership or tend to global competition in protected national markets. End customers might keep trust in specific automotive companies or tend to switch to quickly to competitors benefitting from an attack. Hence, attackers can put higher pressures on companies. For realistic scenarios, consistency is checked for all combinations of projections. The aforementioned scenario results from assumptions that there is a significant risk of international attacks targeting impact on large groups of end customers. Based on these scenarios, threats are analysed upstream: Starting from targeted threats (car defects), the product creation process is investigated towards early production stages (delivery of raw material, printing of circuit boards, coding of software packages). Along

this chain, potential structural and behavioural implications on system elements are annotated within the system model and its partial models.

Fault-tolerance is enabled by applying threat mitigation measures to potential threats. More specifically, defence-in-depth means to combine different measures to mitigate single or aggregated threats across all levels of the system architecture. With regard to the exemplary scenario, various measures could be combined in Systems Engineering: From a software engineering perspective, digital signatures for manufacturing data can avoid data breaches. From a mechanical engineering perspective, triggers can be applied to manufacturing execution data for fault detection and additional testing procedures can help to recognize affected products before shipment.

6 Summary and Outlook

The conceptual approach integrates approaches of misuse case modelling, scenario technique, threat modelling and Model Based Systems Engineering. The challenge is to derive inputs for fault-tolerance engineering from system-level to discipline specific levels of the system architecture. Vice versa, risk assessments and mitigation measures have to be aggregated from component to system level. To achieve both top-level requirements, the entire chain from conceptual modelling of scenarios to threats needs to be formalized in a way that systems engineers and developers are supported on their corresponding interaction levels. An application case is presented as a first validation in laboratory environment. The conceptual approach will be used to contribute to methodological frameworks, modelling languages and computational tools in future.

References

1. Potiron, K., El Fallah, S.A., Taillibert, P.: From Fault Classification to Fault Tolerance for Multi-agent Systems. Springer-Briefs in Computer Science. Springer, London (2013)
2. Isermann, R.: Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance. Springer, Heidelberg (2006)
3. Carnegie Mellon University: Systems Security Engineering Capability maturity Model: SSE-CMM Model Description Document (1999)
4. Fitzgerald, J., Larsen, P.G., Verhoef, M.: From embedded to cyber-physical systems: challenges and future directions. In: Fitzgerald, J., Larsen, P.G., Verhoef, M. (eds.) Collaborative Design for Embedded Systems: Co-modelling and Co-simulation, vol. 138, pp. 293–303. Springer, Berlin (2014)
5. Sadeghi, A.-R., Wachsmann, C., Waidner, M.: Security and privacy challenges in industrial internet of things. In: Proceedings of the 52nd Annual Design Automation Conference. ACM, New York, pp. 1–6 (2015)
6. Lee, E.A.: Cyber physical systems: design challenges. In: 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, pp. 363–369. IEEE Computer Society, Los Alamitos (2008)
7. Gräßler, I., Pöhler, A., Pottebaum, J.: Creation of a learning factory for cyber physical production systems. Procedia CIRP **54**, 107–112 (2016). https://doi.org/10.1016/j.procir.2016.05.063

8. Frazzon, E.M., Hartmann, J., Makuschewitz, T., et al.: Towards socio-cyber-physical systems in production networks. In: Cunha, P.F. (ed.) 46th CIRP Conference on Manufacturing Systems 2013, vol. 7, pp. 49–54 (2013)
9. Nguyen, P.H., Ali, S., Yue, T.: Model-based security engineering for cyber-physical systems: a systematic mapping study. Inf. Softw. Technol. **83**, 116–135 (2017). https://doi.org/10.1016/j.infsof.2016.11.004
10. Deyter, S., Gausemeier, J., Kaiser, L., et al.: Modeling and analyzing fault-tolerant mechatronic systems. In: Norell Bergendahl, M., Grimheden, M., Leifer, L., et al. (eds.) Design Has Never Been This Cool: ICED 09, The 17th International Conference on Engineering Design, pp. 55–66. Design Society, Glasgow (2009)
11. Kolberg, E., Reich, Y., Levin, I.: Express engineering change management. In: Giess, M.P., Goh, Y.M., Lian Ding, L., et al. (eds.) ICED 07, The 16th International Conference on Engineering Design. Design Society (2007)
12. Rostami, M., Koushanfar, F., Rajendran, J., et al.: Hardware security: threat models and metrics. In: 2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 819–823. IEEE Press, Piscataway (2013)
13. Isaksson, S., Ritchey, T.: Protection against sabotage of nuclear facilities: using morphological analysis in revising the design basis threat. Adaptation of the original paper 2010. In: 44th Annual Meeting of the Institute of Nuclear Materials Management (2003)
14. Schumacher, M., Fernandez-Buglioni, E., et al.: Security Patterns: Integrating Security and Systems Engineering. Wiley (2006)
15. Shostack, A.: Threat Modeling: Designing for Security. Wiley, Indianapolis (2014)
16. Walden, D.D., Roedler, G.J., Forsberg, K., et al. (eds.): Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities; INCOSE-TP-2003-002-04, 4th edn. Wiley, Hoboken (2015)
17. VDI: VDI 2206 Design methodology for mechatronic systems (VDI 2206) (2004)
18. Gräßler, I., Hentze, J., Bruckmann, T.: V-models for interdisciplinary systems engineering. In: Proceedings of the DESIGN 2018 15th International Design Conference. Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Croatia, The Design Society, Glasgow, UK, pp. 747–756 (2018)
19. Börjeson, L., Höjer, M., Dreborg, K.-H., et al.: Scenario types and techniques: towards a user's guide. Futures **38**(7), 723–739 (2006). https://doi.org/10.1016/j.futures.2005.12.002
20. Gräßler, I., Hentze, J., Scholle, P.: Enhancing systems engineering by scenario-based anticipation of future developments. In: SoSE and Cyber Physical Systems (CPS), From Academia to Application and Back: 11th Systems of Systems Engineering Conference (SoSE). IEEE, Piscataway (2016)
21. Fitzgerald, J.: Developing & ranking threat scenarios. EDPACS **6**(3), 1–5 (1978). https://doi.org/10.1080/07366987809449432
22. Lotz, V.: Threat scenarios as a means to formally develop secure systems. J. Comput. Secur. **5**(1), 31–67 (1997)
23. Kim, Y.-G., Cha, S.: Threat scenario-based security risk analysis using use case modeling in information systems. Secur. Comm. Netw. **5**(3), 293–300 (2012). https://doi.org/10.1002/sec.321
24. Young, W.T., Memory, A., Goldberg, H.G., et al.: Detecting unknown insider threat scenarios. In: 2014 IEEE Security and Privacy Workshops: Proceedings, 17–18 May 2014, San Jose, California, USA. Conference Publishing Services, pp. 277–288. IEEE Computer Society, Los Alamitos (2014)
25. Roudier, Y., Apvrille, L.: SysML-Sec: a model driven approach for designing safe and secure systems. In: Hammoudi, S. (ed.) MODELSWARD 2015: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development, pp. 655–664. IEEE, Piscataway (2015)

26. Apvrille, L., Roudier, Y.: Towards the model-driven engineering of secure yet safe embedded systems. Electron. Proc. Theor. Comput. Sci. **148**(4), 15–30 (2014). https://doi.org/10.4204/EPTCS.148.2
27. Andress, J.: The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice, 2nd edn. The Basics. Elsevier Science, Burlington (2014)
28. Røstad, L.: An extended misuse case notation: including vulnerabilities and the insider threat. In: 12th Working Conference on Requirements Engineering (REFSQ 2006): Foundation for Software Quality (2006)
29. Gräßler, I., Pottebaum, J., Scholle, P.: Integrated process and data model for agile strategic planning. In: Vajna, S. (ed.) 11th International Workshop on Integrated Design Engineering (2017)
30. Geismann, J., Gerking, C., Bodden, E.: Towards ensuring security by design in cyber-physical systems engineering processes. In: Kuhrmann, M., O'Connor, R.V., Houston, D. (eds.) Proceedings of the 2018 International Conference on Software and System Process - ICSSP 2018, pp. 123–127. ACM Press, New York (2018)
31. Völter, M., Stahl, T., Bettin, J., et al.: Model-Driven Software Development: Technology, Engineering, Management, 1 edn. Wiley Software Patterns Series. Wiley, s.l. (2013)