Integrating Security Considerations Into a Safety Case Construction

Elena Troubitsyna
Åbo Akademi University
Tuomionkirkontori 3, 20500 Turku, Finland
e-mail: Elena.Troubitsyna@abo.fi

***Abstract*— Wide-spread reliance on networking in modern safety-critical control systems makes security increasingly interwoven with safety. Hence, we need novel methodologies integrating security consideration into the process of system development and safety case construction. Safety case is a structured argument justifying system safety. In this paper, we propose an approach that relies on the systems-theoretic analysis to construct security-aware safety cases. We define a number of generic patterns facilitating definition of security-aware safety cases. Our approach allows the developers to analyse the mutual interdependencies between safety and security in the design of networked control systems. It provides the engineers with a systematic top-down method for deriving constraints that should be imposed on the system and software behavior to guarantee safety in the presence of accidental and malicious faults.**

Keywords-*safety case; systems-theoretical approach; controlling software; security; integrated analysis*

I. INTRODUCTION

Traditionally safety-critical systems have been considered as closed systems that should ensure safety despite (accidental) components faults [1]. However, increasing openness and reliance on networking has introduced security attacks, i.e., malicious faults, as an important factor to be analyzed in the process of system development and verification [2].

Since safety and security are often considered as separate fields, there is a lack of integrated approaches that support the holistic analysis of software-intensive systems that can guarantee safety in presence of both malicious and accidental faults [1]. However, recent research experiments have demonstrated, e.g., that cars security vulnerabilities allow to remotely override safety functions and take control over break and steering [3]. Therefore, there is a clear need for the approaches that provide the developers with an integrated view on system safety and security.

In this paper, we propose an approach to integrating the security consideration into the process of safety case construction for networked safety-critical control systems.

Safety case is a structured argument about system safety [4]-[8]. Often, it is defined using Goal Structuring Notation [9]. While constructing a safety case, we explicitly define the links between top-level goal of achieving system safety and the satisfaction of constraints that should be imposed on the system design to achieve it.

To derive safety and security constraints required for achieving safety, we propose to employ the systems-theoretic analysis [10]. Systems theory considers the problem of ensuring safety as a control problem and as such, provides us with a more inclusive model of accident causality. Therefore, the systems-theoretic perspective supports an integrated consideration of safety and security constraints that are essential in designing networked control systems.

In this paper, we demonstrate how an application of the systems-theoretic analysis allows us to define the main classes of causes that might lead to unsafe behavior and define the corresponding safety goals. By top-down decomposition of such goals, we define safety and security constraints that should be imposed on the system design to guarantee safety. We define the patterns of safety case fragments that allow us to justify safety in presence of both accidental and malicious faults.

We believe that an application of the proposed approach enables holistic analysis of safety and security interdependencies and facilitates construction of safe networked control systems.

The paper is structured as follows: in Section II, we introduce the notion of the safety case and the Goal Structuring Notation. In Section III, we describe the principles of systems-theoretical analysis. In Section IV, we present our approach to constructing security-aware safety cases using systems theory. In Section V, we overview the related work. Finally, in Section VI, we discuss the proposed approach.

II. SAFETY CASES

A safety case is "a structured argument, supported by a body of evidence that provides a convincing and valid case that a system is safe for a given application in a given operating environment" [4] [5].

The construction, review and acceptance of safety cases are the important steps in safety assurance process of safety-critical systems. Several standards, e.g., ISO 26262 [6] for the automotive domain, EN 50128 [7] for the railway domain, and the UK Defense Standard 00-56 [8], prescribe production and evaluation of safety (or more generally assurance) cases for certification of such systems.

 ISBN: 978-1-61208-492-3

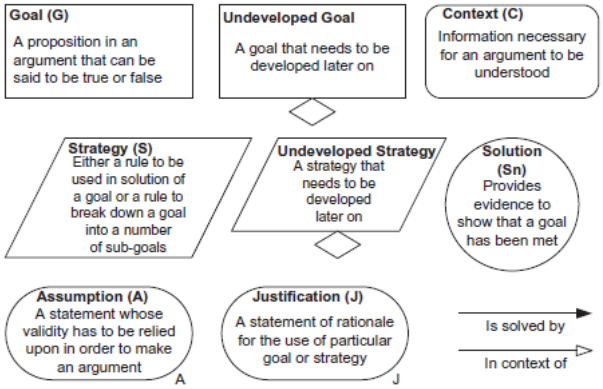

Figure 1. Basic elements of GSN.

A safety case can be defined textually or graphically. Currently, Goal Structuring Notation (GSN) – a graphical notation proposed by Kelly [9] – is getting increasingly popular for describing safety case. GSN aims at a graphical representation of safety case elements as well as the relationships that exist between these elements. The main building blocks of GSN are shown in Figure 1.

Essentially, a safety case constructed using GSN consists of goals, strategies and solutions. Here goals are propositions in an argument that can be said to be true or false (e.g., claims of requirements to be met by a system). Solutions contain the information extracted from analysis, testing or simulation of a system (i.e., evidence) to show that the goals have been met. Finally, strategies are reasoning steps describing how goals are decomposed and addressed by sub-goals. Thus, a safety case constructed in GSN presents a decomposition of the given safety case goals into the sub-goals until they can be supported by the direct evidence (a solution). It also explicitly defines the argument strategies, relied assumptions, the context in which goals are declared, as well as justification for the use of a particular goal or strategy.

The elements of a safety case can be in two types of relationships: ‘‘Is solved by’’ and ‘‘In context of’’. The former is used between goals, strategies and solutions, while the latter links a goal to a context, a goal to an assumption, a goal to a justification, a strategy to a context, a strategy to an assumption, a strategy to a justification.

A typical high-level structure of the safety case is shown in Figure 2. The high-level goal **G1** contains the proposition that the system is safe. The strategy **S1** is to decompose top-level goal into lower level subgoals **G2-G$_{N+1}$** aiming at demonstrating that each individual hazard has been mitigated. The safety case is valid under the assumption **C1** that all hazards have been identified.

Usually, to achieve completeness of hazard identification the developers rely on safety analysis, e.g., fault trees, Failure Modes and Effect Analysis, etc. However, Leveson [10] points out that such techniques rely on linear causality models and lack the power to exhaustively analyse

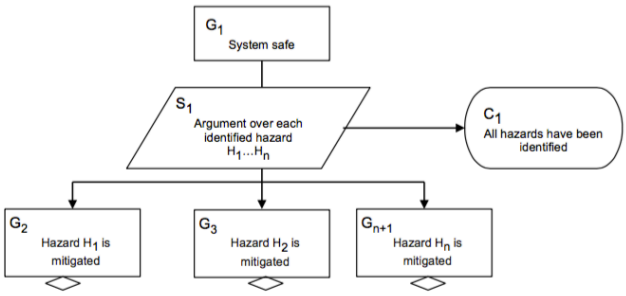

Figure 2. High-level safety case.

hazardous behaviour in complex software-intensive systems. She argues that we need to rely on systems-theoretic approaches to guarantee safety of such complex systems.

In complex software-intensive systems, hazards might be caused not only by accidental (i.e., non-malicious) component failures but also security failures caused by attacks on system network infrastructure, design errors, unforeseen component interactions, etc. Therefore, we need the integrated systems-theoretic approaches that allow us to identify the strategy for protecting the services and functions that are essential for ensuring system safety in presence of disruptions of various natures.

Next, we present a systems-theoretic approach to integrated reasoning about safety of complex networked systems that are subjects of accidental and malicious faults.

III. SYSTEMS-THEORETIC APPROACH

Systems theory establishes foundations for engineering complex systems [11]. It provides a more inclusive model of accident causality called STAMP – System-Theoretic Accident Model and Processes [10]. STAMP envisions losses as resulting from interactions among humans, physical system components and the environment that lead to the violation of safety constraints. The main difference between STAMP and the traditional approaches to safety is that it shifts the focus from preventing failures to enforcing safety constraints on system behavior.

To illustrate the main principles of a systems-theoretic approach, let us consider let us consider a generic control system. A control system is a reactive system with two main entities: an environment and a controller. The environment behaviour evolves according to the involved physical processes and the control signals provided by the controller. The controller monitors the behaviour of the plant and adjusts it to provide intended functionality and maintain safety. The control systems are usually cyclic, i.e., at periodic intervals they get input from sensors, process it and output the new values to the actuators. The general structure of a control system is shown in Figure 3.

The controller is a hierarchical control structure that constraints the system behavior. Each layer of it enforces the required constraints on the behavior of the components at

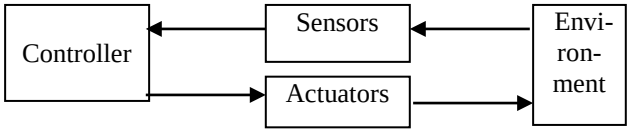

Figure 3. A general structure of a control system.

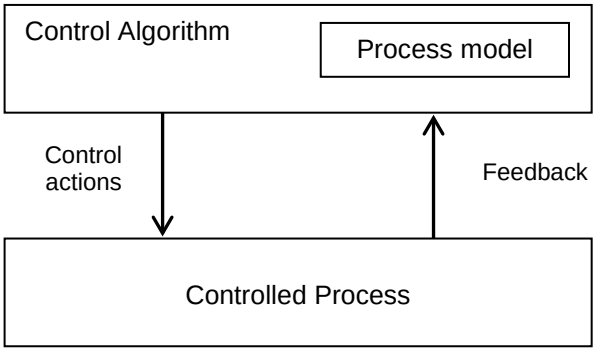

Figure 4. Systems-theoretic view on control system.

the next lower level. The control loops operate between the layers. To achieve safety, we should guarantee that, under the hostile environmental conditions and in presence of accidental and malicious faults, the control actions prevent hazard occurrence.

In systems and control theory, the controller contains a model of the process that it controls. Such a model serves as a basis for defining the necessary control actions, as shown in Figure 4. Hazards often occur as a result of inconsistencies between the controller's model of the controlled process and the actual process state.

By applying systems-theoretic analysis and analyzing the general structure of a control system, we can observe that safety can be violated due to three types of causes

- Controller cannot built a correct model of the process because the measurements provided by the sensors are invalid
- Controller has the correct model of the process but there is a logical error in implementing correct control actions
- The actuator fails to correctly implement the control actions.

This observation allows us to refactor our generic safety case pattern presented in Figure 1 as shown in Figure 5. It reflects the systems-theoretic approach to ensuring safety and establishes a systematic way for construct the safety case by further decomposition of subgoals **G2 –G4.**

In the next section, we propose a systematic approach to analyzing how the accidental and malicious faults introduce inconsistencies into the controller's model of the process, distort the logic of the controller or prevent correct implementation of the controller actions.

IV. SECURITY-INFORMED SAFETY CASES

Let us again consider the generic control cycle presented in Figure 3. For many control systems, safety can be formulated as the following proposition:

The value of critical parameter p always remains within safe boundaries.

To achieve this safety goal, we need to systematically analyse the causes that can introduce hazardous deviations in the controller's model of the process controlling *p* and the actual state of *p*.

To build the corresponding model of the process, the controller relies on the measurement of *p* provided by the corresponding sensor. Therefore, the first condition for ensuring accuracy of the controller's model is validity of sensor's reading.

The sensor's readings can be distorted due to accidental faults of the sensor or security attacks. If the sensor fails and the controller does not detect it, then it starts to rely on wrong data. Hence, to guarantee safety, we should ensure that the sensor health is monitored and upon detection of failure the controller starts to rely on alternative reliable sources of measurement of *p*.

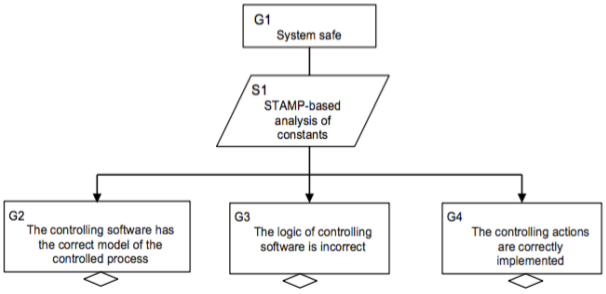

Figure 5. Systems-theoretic approach to safety case.

Typically, safety-critical control systems contain some form of redundancy. For instance, there might be hot or cold spare sensors. In the first case, the controller simply switches to obtaining readings from the spare sensor without any disruption in measurement provisioning. In the second case, a certain time interval is required to activate the cold spare. The system design should ensure that the time period required for the reconfiguration is sufficiently short, i.e., it would not introduce dangerous deviations in the controller's model of the process while the measurements are not available.

The controller might also obtain the invalid measurements of *p* due to security attacks. In the context of our control loop, it is relevant to consider the following security failures:

- spoofing the identity of sensor and
- tampering sensor data by attacking the communication channel between the sensor and the controller.

By spoofing the sensor identity the attacker can supply the controller with the deliberately wrong measurements of *p*. They can "trick" the controller into thinking that the value

 ISBN: 978-1-61208-492-3

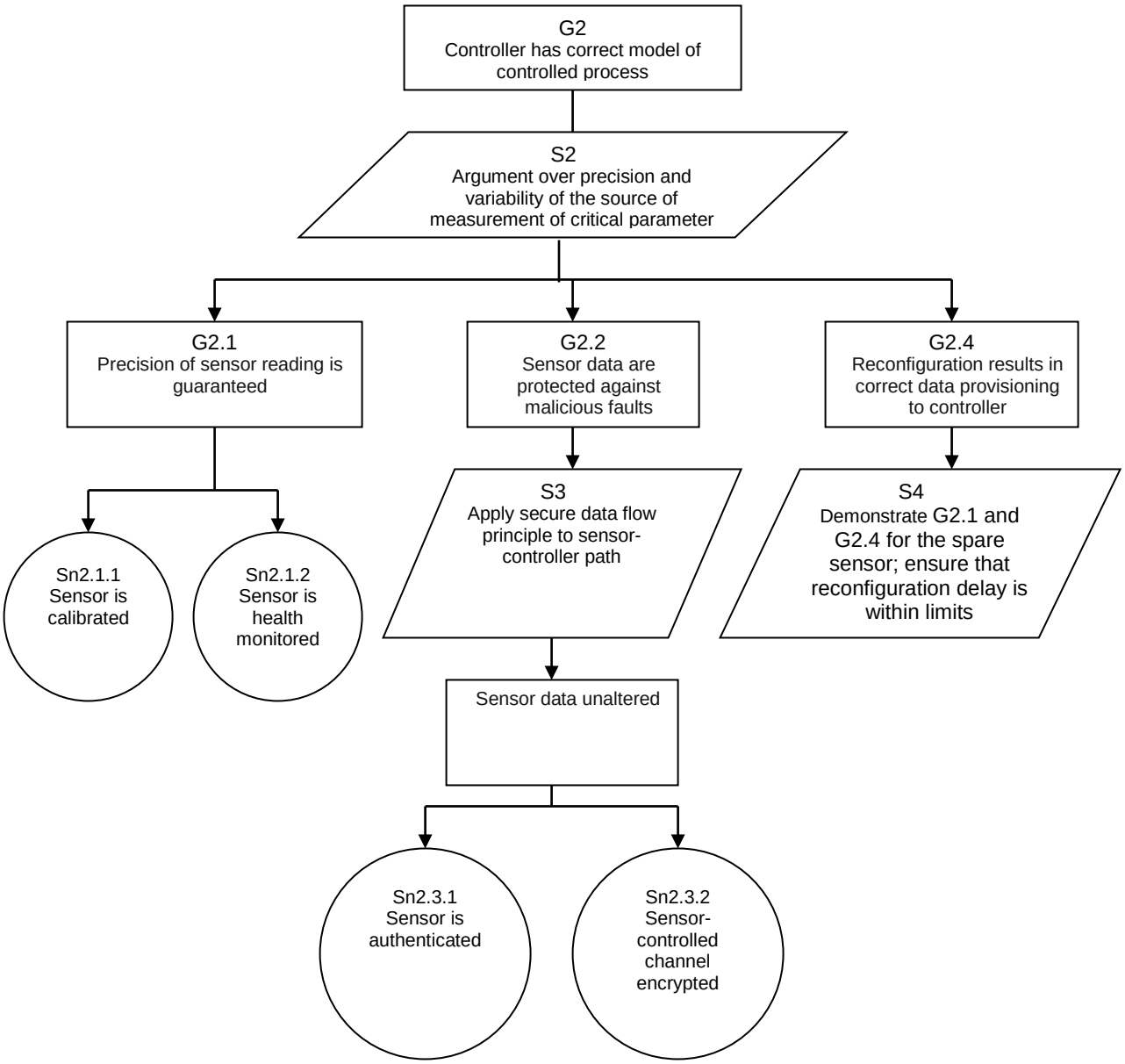

Figure 6. Pattern for **G2** decomposition.

of the controlled parameter is well within safety limits. This is a dangerous deviation of the controller's model of the process. Hence, it is very likely to result in hazard occurrence due to controller inability to issue the correct control commands required to maintain safe value of *p*. Tampering with sensor data has the same effect.

The analysis above demonstrates the direct impact of security on safety. The systems-theoretic approach allows us to identify the strategy for protecting the systems. Namely, we should guarantee that the source of measurement of *p* is authenticated and the communication link between the sensor and the controller is encrypted, i.e. does not allow for unauthorized data alternations. In the similar way, to ensure that the sensor failures are reliably detected, we need to guarantee that the health monitoring data is not tampered with.

We can also demonstrate that safety-security interdependencies are sometimes conflicting and require trade-offs.

To ensure security, the design of software-intensive systems typically follows multi-level secure systems principle introduced by La Padula and Bell [12]. Often the designers consider two security levels: high, meaning highly sensitive or highly trusted, and low, meaning less sensitive or less trusted. When the trusted components of the system interact with the untrusted parts, one has to ensure that there is no indirect leakage of sensitive information from the trusted to untrusted part. Usually it is defined as no "down-flow" policy. Such a security requirement is commonly

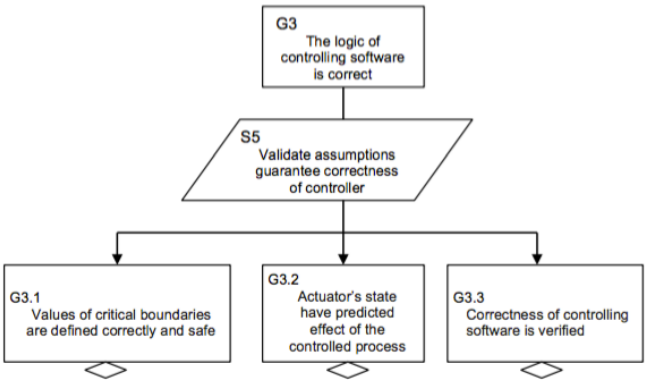

Figure 7. Pattern for **G3** decomposition.

called *secure information flow*.

Let us consider how this principle is implemented in our case. As a result of the analysis above, we have derived the requirements that ensure no down flow policy for the sensor providing the measurements of *p*. Now, let us consider the case when the sensor providing the measurements of *p* has failed and the controller has to use the alternative sources of measuring *p*.

For the spare sensor, the system might not have the authentication and encryption procedure implemented and hence switching to the use of the alternative sensor would break the secure information flow policy. By preventing the use of measurements provided by the spare sensor, we leave the controller without the feedback required to build the adequate model of the controlled process. On the other hand, the use of unauthenticated sensor and unencrypted channel introduces security vulnerability. It is clear that we should resolve this conflict. For instance, we might run spare sensor authentication upon reconfiguration and require to use the encryption once the spare sensor becomes the primary source of measurements of *p*.

The system-theoretic analysis allows us to construct the corresponding part of the safety case, as shown in Figure 6.

Now, let us discuss the constraints that should be imposed on the system to ensure that the goal **G3** is achieved. Essentially, we have to verify that the controller actions maintain the safety invariant *"p is within safety boundaries"*. To verify this we have to introduce a number of assumptions.

The first class of assumptions explicitly states the impact of the actuator state on the value of the controlled parameter. Let us explain it by an example. Assume that the controlled parameter is a temperature *t*. The actual temperature should be kept within safety boundaries *t_min_crit* and *t_max_crit*.

The temperature is controlled by switching on and off the heater. The assumptions that we make is that when the heater is switched on the temperature is increasing. Correspondingly, when the heater is switched off the temperature is decreasing.

Another class of assumptions that we need to introduce deals with the inertia of the controlled physical process and relies on the cyclic behavior of the system. Since the controller receives the measurements of the controlled parameter once per control cycle, it needs to issue the control actions changing the state of the actuator before the critical boundaries are reached.

To demonstrate that the actual value of the parameter always remains within safety limits, we need to constrain Δ - the maximum possible imprecision of the parameter in the process model as well as Δ_{max_cycle} – the maximum possible change of the parameter per cycle.

The state of the actuator should be changed to the one that leads to the increase of the parameter at *P_min*, which is greater than *p_min_crit* at least for the sum of Δ and Δ_{max_cycle}. The similar condition is imposed on P_{max}. Under these assumptions, we can verify that the controlling software maintains safety invariant, i.e., we can argue for achieving the goal **G3**. The corresponding fragment of the safety argument is shown in Figure 7.

Next, we investigate the constraints that should be imposed on the system to justify achieving goal **G4**. It is obvious that if the actuator fails and its failure remains undetected then it directly leads to failure to implement the commands of the controller in the correct way. Therefore, we should guarantee that the failures of the actuator are reliably detected and the system is put in a safe non-operational state upon it.

Now, let us consider the security-related constraints that should be satisfied to guarantee achieving goal **G4**. Even though the controller could have issued the correct control commands, due to spoofing controller identity or tampering commands the actuator might receive the incorrect settings that might breach safety. Therefore, we have to enforce secure data flow policy on the communication between the controller and the actuator as well. The corresponding fragment of the safety case is shown in Figure 8.

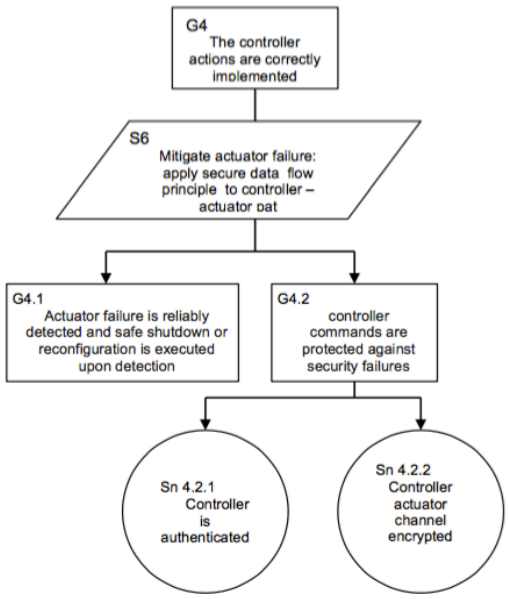

Figure 8. Pattern for **G4** decomposition.

We can summarize the proposed methodology as follows:

1. Apply systems-theoretic approach to analyse how the controller builds the process model.
2. Define the top-level safety goal and identify critical parameters that should be monitored and controlled. Define safety conditions over these parameters.
3. Create an architectural model of the system and identify
    a. the components involved into providing input to the controller allowing it to build the process model (sensors)
    b. the components responsible for implementing controller actions (actuators)
4. For the identified components analyse the impact of failures and define the mitigation actions required to achieve safety goals. Construct the corresponding fragments of the safety case
5. Analyse data flow and define security constraints guaranteeing secure data flow policy for monitoring the critical parameters and implementing controller actions. Construct the corresponding fragments of the safety case
6. Derive the constraints required to verify correctness of the controller logic. Construct the corresponding fragment of the safety case.

V. Related Work

Currently, the problem of integrated analysis of safety and security is receiving significant research attention. Schmittner at al. propose an approach that adapts Failure Mode and Effect and Criticality Analysis to address safety failures [13]. The work demonstrates how to take into account the motives of the intruder as well as costs and complexity of exploiting vulnerabilities. The approach proposed by Schmittner et al. can be used as an input for the safety case construction technique presented in this paper.

The approach relying of the integration of safety consideration into fault tree analysis has been proposed by Steiner and Liggesmeyer. The approach provides the engineering with a structured way to discover and analyse security vulnerabilities that have safety implications. This work complements the systems-theoretical approach to construction of the safety cases proposed in this paper.

Formal approaches proposed to study security and safety interactions typically focus on finding conflicts between safety and security requirements [15]. The majority of the approaches demonstrate how access control rules contradict safety requirements. In our approach, we do not contrapose safety and security but rather derive the security and safety constraints in top-down manner based on the safety cases. The advantage of our approach lies in its ability to capture the dynamic nature of safety and security, e.g., resulting from the reconfiguration required to achieve fault tolerance.

VI. Conclusions

In this paper, we have proposed a systematic approach to construction of security-aware safety cases. In our approach, derivation of safety and security constraints proceed hand-in-hand with safety case construction. The use of systems-theoretic reasoning allows us to derive the constraints required for providing arguments for safety case in a disciplined top-down way. Such an approach supports an integrated reasoning about safety and security that facilitates analysis of requirements interdependencies and explicit identification of trade-offs required to achieve safety in presence of both malicious and accidental failures.

In our future work, we are planning to validate the proposed approach in a number of industrial case studies as well as provide an automated tool support linking systems-theoretic analysis and safety case construction.

References

[1] R. Bloomfield, K. Netkachova, and R. Stroud, “Security-Informed Safety: If It's Not Secure, It's Not Safe”. Workshop on Software Engineering for Resilient Systems, LNCS 8166, Springer, Sept. 2013, pp. 17-32. DOI 10.1007/978-3-642-40894-6_2.

[2] W. Young and N.G. Leveson, “An integrated approach to safety and security based on systems theory”. Communication of ACM 57(2), 31-35, 2014, DOI 10.1145/2556938.

[3] Online http://www.bbc.com/news/technology-35841571. Accesed 02.04.2016.

[4] T. Kelly and J. McDermid, “Safety case construction and reuse using patterns”. 16th International Conference on Computer Safety, Reliability and Security (SAFECOMP’97), Springer-Verlag, Sept. 1997, pp. 55–96, doi: 10.1007/978-1-4471-0997-6_5.

[5] P. Bishop and R. Bloomfield, “A methodology for safety case development”, in: Safety-Critical Systems Symposium, Springer-Verlag, Feb.1998, pp.10-16, , doi: 10.1008/243-1-3382-06577-5_6.

[6] International Organization for Standardization, ISO 26262 Road Vehicles Functional Safety, 2011.

[7] European Committee for Electrotechnical Standardization (CENELEC), EN 50128 Railway Applications – Communication, Signalling and Processing Systems – Software for Railway Control and Protection Systems, 2011.

[8] Defence Standard 00-56, UK Ministry of Defence. 00-56 Safety Management Requirements for Defence Systems, 2007.

[9] T.P. Kelly, “Arguing safety -- a systematic approach to managing safety cases”, PhD Thesis York University, UK, 1998.

[10] N. Leveson, “Engineering a safer world: Systems thinking applied to safety”. In MIT Press. 2011.

[11] P.Checkland, Systems Thinking, Systems Practice. Wiley, 1981.

[12] D.E.Bell and L.J LaPadula.. "Secure Computer Systems: Mathematical Foundations", Journal of computer Security, 1996, 4:239-263.

[13] C. Schmittner, T. Gruber, P.P. Puschner, and E. Schoitsch, “Security application of failure mode and effect analysis (FMEA)”. In: SAFECOMP 2014. LNCS 8666, Springer, Sept. 2014, pp. 310-325.

[14] M. Steiner and P. Liggesmeyer, “Combination of Safety and Security Analysis - Finding Security Problems That Threaten The Safety of a System”. Workshop on Dependable, Embedded and Cyber-physical Systems, Toulouse, France, 2013, pp.7-14.

[15] P. Bieber and J. Brunel, “From Safety Models to Security Models: Preliminary Lessons Learnt”, Workshops at SAFECOMP 2014, pp.269—281, DOI 10.1007/978-3-319-10557-4_

 ISBN: 978-1-61208-492-3