HAL
open science

Modeling and Analysis of Safety-Critical Cyber Physical Systems using State/Event Fault Trees

Michael Roth, Peter Liggesmeyer

To cite this version:

Michael Roth, Peter Liggesmeyer. Modeling and Analysis of Safety-Critical Cyber Physical Systems using State/Event Fault Trees. SAFECOMP 2013 - Workshop DECS (ERCIM/EWICS Workshop on Dependable Embedded and Cyber-physical Systems) of the 32nd International Conference on Computer Safety, Reliability and Security, Sep 2013, Toulouse, France. pp.NA. hal-00848640

HAL Id: hal-00848640

https://hal.science/hal-00848640v1

Submitted on 26 Jul 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Modeling and Analysis of Safety-Critical Cyber Physical Systems using State/Event Fault Trees

Michael Roth, Peter Liggesmeyer

Department of Software Engineering: Dependability, University of Kaiserslautern, Germany
{michael.roth|liggesmeyer}@cs.uni-kl.de

Abstract. Modern cyber physical systems (CPSs) are becoming more and more vulnerable to security related attacks, due to the growing number of interconnectivity and standardized communication channels. This evolution make the traditional approaches considering the safety and security domains as two disjunctive areas obsolete. In this paper we propose state/event fault tree for modeling and analyzing the safety and the security aspects of CPSs in a common model. To evaluate our approach, we apply it on a case study of a tire pressure monitoring system.

Keywords: reliability modeling, risk analysis, attack trees, fault tree, State/Event Fault Tree

1 Introduction

For ensuring the safety of technical systems, fault trees (FTs) are an established methodology. Through their logical gates and hierarchical decomposition they are intuitively understandable. Due to their ability to capture qualitative and quantitative analysis aspects they can be seen as state of the art in safety analysis.

Especially in the field of cyber physical systems (CPSs), where numerous interfaces come into account, FTs are not well suited because of their inability to deal with security threats. However, some approaches can be found which transfer FTs from the safety to the security domain, but they are still unable to model the interdependencies of the two worlds. The Stuxnet worm for example shows that the separate view of both fields is no longer tolerable. Nicely visible examples for such an adoption are attack trees (ATs) [2] and fault trees for security [9]. In the area of software-controlled systems these tree based approaches have some additional crucial disadvantages. They are not able to deal with temporal aspects and it is not possible to do a quantitative analysis of them in case of statistical depend basic events. In [6] Kaiser et al. propose a new technique to combine fault trees with explicit state/event semantics by using a graphical notation similar to state charts.

In our work we propose a method which extends these state/event fault trees (SEFTs) with an attacker model to derive a potent model to deal with safety and security equally.

2 Related Work

Fault tree analysis is a widely accepted methodology in the field of safety and reliability engineering. Since H. R. Watson [5] introduced the technique in the 1960's, FTs were consequently improved and matched on the changing requirements. Schneier adapted the tree structured notation to so called ATs to describe security risks for a system [2]. These trees provide a formal method to describe how varying attacks, modeled as leaf nodes of the tree, harm the overall security property of a system. This property is modeled as the root node of the tree, connected via logical gates with its leafs. An extension of ATs is made in [12]. Fovino et al. published a combining approach to model security risks together with an attacker in one model. They show how attacker operations could be combined with vulnerabilities and assertions in a flat tree structure to identify the attacker's impacts on the system for a qualitative analysis. In [7] Piètre-Cambacédès et al. deal with interdependencies of safety and security in the field of Markov modeling. They introduce an easily understandable notation for Markov processes similar to a tree structure. With their integrated models it is possible to model dependent events, as it is necessary in warm-standby situations. The analysis of the model can be done by compiling it into a markov process (MP). As an equivalent to the failure rates and the mean time to failure (MTTF), they introduced a measure for the security domain, called attack rate. This rate can be determined with its reciprocal, the mean time to success (MTTS). With the use of these constant rates it is possible to use state of the art analyzing tools for Markov processes.

All the above mentioned approaches are not able to consider a system in conjunction with a potent adversary model. The work of Benenson et al. [10] defines a model of an attacker which is able to compromise wireless sensor networks. Their idea is a three-dimensional view of potential intervention options (safety, liveness and information-flow) of the attacker to compromise network nodes. With useful combinations of different levels of the dimensions it is possible to define most likely attacker profiles w. r. t. sensor networks. In [11] Vigo introduces an attacker model especially for CPSs which is able to exploit both the cyber weakness and the physical weakness of the system. The ADVISE approach [13] couples an attacker model with an attack graph, which describes the correlation of possible attacks to the system. By the use of well specified attack properties (payoff, costs, noticeability, ...), the attackers preferences and his knowledge an algorithm calculates a decision-tree which indicates the most likely path through the graph for a special adversary profile.

According to our best knowledge, there exists no technique which provides the possibility to couple attacker models with an easily handleable tree-based system model to perform a domain crossing safety and security analysis.

3 State/Event Fault Trees

SEFTs bridge the gap between easy understandable FTs and powerful state charts and bring them together into one model. This makes it possible to model

deterministic state spaces and probabilistic failure behavior with the visualization power of original FTs. Fig. 1a) gives a brief overview of the modeling elements of SEFTs. To adopt the model to the architecture of a real world system, SEFTs provide a component concept where components(I) can communicate with each other and failure propagation is facilitated with in-ports(II) and out-ports(III). In SEFTs, the temporal dependencies are modeled within the components by the use of state charts, where the state changes can be triggered by exponentially distributed probabilistic events(IV), deterministic events(V) and triggered events(VI). These triggered events can be seen as externally controlled transitions. All events can be guarded by states(VII). This means that a guarded event is only able to fire if the connected state is active. States and events have to be connected by using so called temporal connections(VIII). In contrast, causal dependencies of the component's states and events are modeled, as typical in fault trees, with gates(IX) using causal connections(X). The functional range of these gates is thereby far beyond the functional range of logical gates of standardized fault trees. There exist pure state gates for the linkage of various states and pure event gates which are able to link different events. Finally, there exists mixed gates for linking states and events. To connect the gates with the component's states and events, their in-ports and out-ports are further refined into typed ports for states (II-I, III.I) and events (II-II, III.II). The whole gate dictionary can be found in [6].

We decided to demonstrate the methodology based on an easy understandable example from the chemical domain, even though it provides no safety and security interdependencies. Due to its simplicity it is well suited to explain the functioning of SEFTs. Fig. 1b) shows a SEFT of a reactor's safety circuit. The reactor reaches a safety critical state in case of a defective pressure valve or a defective pressure sensor together with an exceeding of the critical pressure threshold in the reactor. The OR-gate in Fig. 1b) is modeled as a pure state gate, called *OR_State*-gate, with two state inlets for the defect states of the valve and the sensor. The AND-gate is modeled as a mixed gate, a so called *AND_Event_State*-gate, with one event inlet and one state inlet. The gate's outlet event fires if the state connected to the state inlet is active and the event connected to the event inlet fires. Here, a temporal dependency is nicely shown by the fact that one of the safety-related devices has to fail before the critical limit is exceeded.

SEFTs are especially designed to do a quantitative analysis, by translating them into extended deterministic stochastic petri nets (eDSPNs). To deal with SEFTs, the ESSaRel modeling tool [4] has been developed which makes it easy to model the trees and convert them into eDSPNs. Subsequently, these nets can be analyzed by a tool called TimeNet [3]. The therein realized analysis methods are well investigated and can be seen as state of the art techniques like steady state analysis or Monte Carlo simulation. So questions like "what is the probability of a place to be in a special state after a given time?" can be answered.

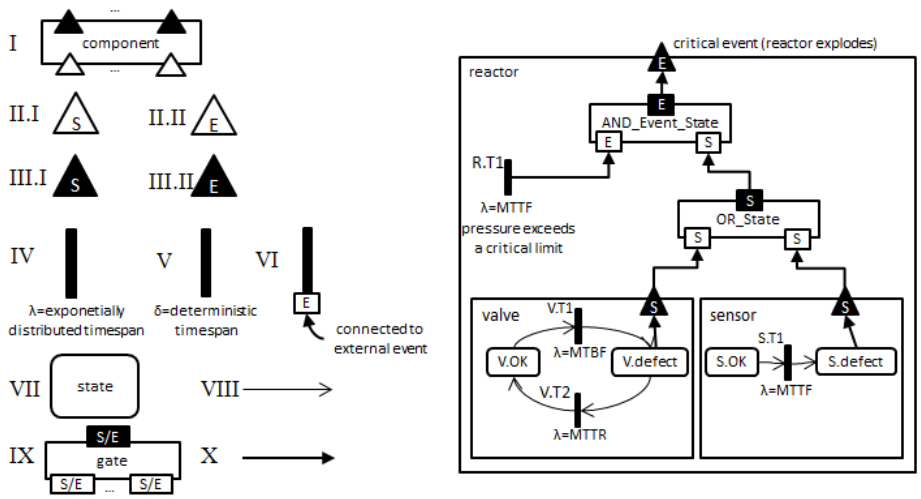

Fig. 1. a) Modeling elements of SEFTs b) Safety system of a reactor modeled as a SEFT

4 Modeling Vulnerabilities

A CPS is prone to physical and cyber-space attacks, due to the adversary is able to directly attack the physical components or attack them via their exchanged messages. This means that both software as well as hardware can be concerned in the attackers actions. As shown in [8] SEFTs are equally suitable to model software and hardware failures. For this reason we decided to use SEFTs to build cross-domain models. In this section we like to introduce possible attack points in a SEFT based model. Such basic vulnerabilities within a SEFT model can be described as follows:

- **Denial of Service (DoS) of exchanged messages** between components
- **Spoofing** of messages between components
- **DoS of a component** by hinder one or more transitions from switching
- **Bypassing** a component's states. An attacker has the possibility to modify components by changing transitions between states and bring in new ones which can be seen as "shortcuts" between states.
- **Reprogram** the component's state chart. The attacker is able to bring in new states and has the possibility to change the inner structure completely.

All of these vulnerabilities (except the spoofing vulnerability) can be related to both the cyber-space and the physical world. In a SEFT, these attack points have to be modeled using the SEFT syntax. In case of a DoS of a communication channel an attacker has the ability to intercept messages or commands between components, e. g. hinder messages of being received over a wireless channel. That can for example lead to an omission failure of a system's component. It

is not required to have network access to exploit such a DoS vulnerability in a communication channel. For this reason it is a relatively easy to attack systems in this way. Fig. 2a) depicts this DoS-pattern. It is only possible to send a message over a channel when the state at *S1* is active. In case of an attack the *Flip-Flop*-gate is triggered at its *Set* inlet which changes the state outlet of the *Invert*-gate to inactive, what means that a communication is no longer possible. In contrast, the spoofing attack pattern (Fig. 2b) allows the intersperse of messages, e. g. to transfer a receiving component into another state or to make a receiving component believe that everything is fine (this pattern is described further in a case study in Sec. 6). A fake message at event inlet *E2* of the *OR_Event*-gate will make the component *C1* believe that it receives a message from component *C2*. For spoofing attacks it is necessary to get access to the communication layer of the system. A DoS of a component (Fig. 2c) is different to a DoS of a communication channel in the way, that the attacker has access directly to the component and thereby the ability to hinder it from switching into another state. Here it is additionally possible to hinder transitions from switching if they are not triggered from outside over the component's interfaces. The gate structure works similar as in the DoS attack of communication channels. In subfigures d) and e) of Fig. 2 more complex vulnerabilities are given, where an attacker could change the behavior of the component by bypassing state *S2* with the event in-port *E* (d). In e) a reprogramming attack pattern is shown, where a attacker can change the complete behavior of the state chart by a trigger at event port *E*. Due to a better visibility all security related aspects are modeled with dotted lines.

5 Modeling the Cyber-Physical Attacker

To deal with these kind of vulnerabilities it is beneficial to bring a model of the adversary into account. Therefore, we would like to introduce an attack component, representing the cyber-physical attacker, nested in the system's environment and connected via ports to the system's vulnerabilities. Such an attacker can execute various attack steps to reach his goal. We introduce attack steps as subcomponents of the attack component which can be connected to each other through event ports to build logical attack queues (Fig. 3-left side). On the right side of Fig. 3 a proxy for an attack step is depicted, modeled as a state chart. In an attack step's state chart it is possible to choose an exponential distributed timespan (λ) for indicating one attack cycle. The related stochastic event is guarded by the state out-port of an inner component, called *Activation*, which activates the attack step with an event at the *Set* in-port and deactivates it after a given time duration δ or with an event at the *Reset* in-port (e. g. if the attacker reaches his goal, all remaining attacks could be stopped). A third parameter is given by a probability value (γ) to define the success probability of an attack cycle (e. g. probability that a guessed password is correct or not). For quantitative analysis these values can be added by experts, determined with statistics or by coupling both. These probabilistic based out-ports can, for example, be

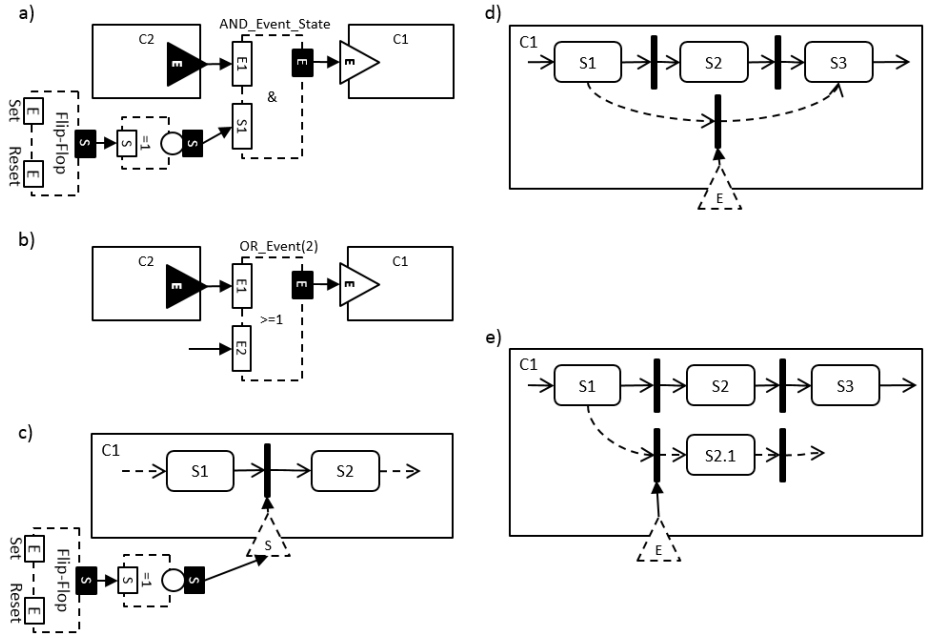

Fig. 2. Vulnerability patterns in SEFTs

used to connected to the possible vulnerabilities of a system or to activate the subsequent attack step(s) in the queue. We tried to model the proxy as simple as possible to reduce the risk of getting lost in an over-detailed attack model. An important advantage of the above introduced method is that SEFTs can analyze stochastically dependent events (in contrast to standardized FTs) which are usually existing in the security domain.

6 Case Study: Tire Pressure Monitoring System

In this section we will show how the presented approach may be applied to a real world system. We decided to choose a wireless, also called direct, tire pressure monitoring system (TPMS) because of its relations between safety and security. Most important, it is a safety critical system processing interfaces that represent weak points in the sense of security. The TPMS consist of 4 tire pressure sensors (TPSs), installed on the inner side of the tire valves, 4 antennas which receive the signals of the sensors, an electrical controlling unit (ECU) for data-processing and a dashboard unit which alerts the driver in case of an not fully inflated tire. The transmission of the pressure is done via a high range signal of 433 MHz. These wireless interfaces make the TPMS vulnerable to attacks, their wide-spread use make them attractive for attackers (TPMSs are installed in every car sold after 2008 in the US and in every new car produced since 2014 in the EU) and their poor security mechanisms are easy to hack.

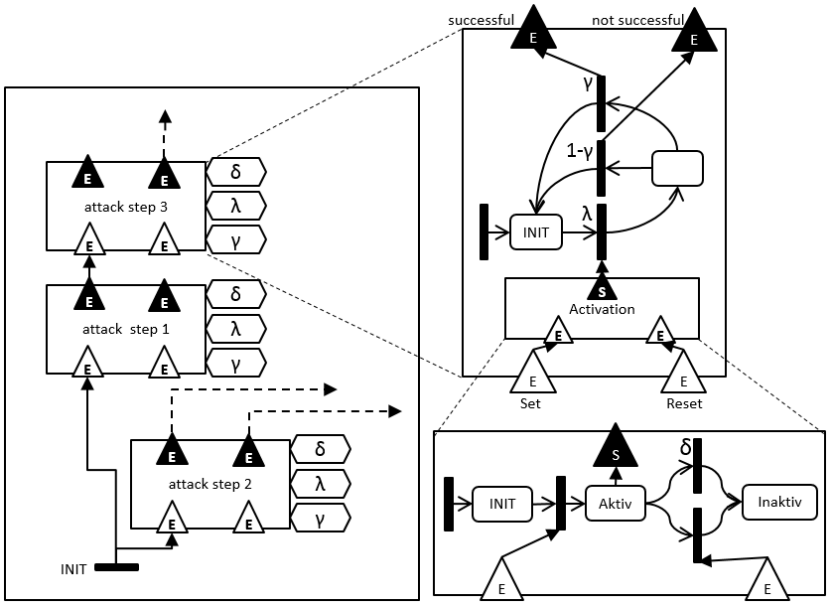

Fig. 3. Attack component with different attack steps

We model the TPMS that is introduced in [1]. It has no encryption method, no input validation and no sequence package number implemented. Its modulation and encoding scheme is easy to figure out and is well investigated in [1]. The ECU accepts all messages from its known identifiers and it is not very hard and no expensive hardware is needed to extract the ID from a received package. In [1] it is described how the eavesdropping range could be increased from usually 10 m up to 40 m. To save battery power the sensors start working at a speed of more than 40 km/h and send a status message at least every 60 s. A possible attack, described in [1], could be the sending of messages indicating less air pressure. Due to the lack of security mechanisms this can affect the driver to stop on the roadside which brings him in a dangerous situation. To determine the probability P_i to receive a message, for extracting the ID, of one sensor of a car its total time within the receiver's range is needed. If we assume that the car is driving with a speed of 120 km/h, it stays at least 2.4 s within the range of max. 80 m (Equ. 1). This results in a probability of 4 % that a sensor sends a status message while it is in the attackers range (Equ. 2). So the total probability to get at least one message of a car's 4 sensors is about 15 % (Equ. 3). We decided to set the success-probability γ to 7.5 %. That leaves the attacker enough time to extract the ID and send a fake message before the car leaves the transmitter's range again. For more information about the TPMS and the real time decoder please refer to [1].

$$rangeTime = \frac{range[m]}{speed[km/h] * 1/3.6} = \frac{80\,m * 3.6}{120\,km/h} = 2.4\,s \tag{1}$$

$$P_i = \frac{rangeTime[s]}{pingTime[s]} = \frac{2.4\,s}{60\,s} = 4\,\%; i = 1, ..., 4 \tag{2}$$

$$P_{total} = 1 - \prod_{i=1}^{4}(1 - P_i) = 1 - 0.96^4 = 15\,\% \tag{3}$$

In Fig. 4a) the SEFT model of the TPMS is depicted. We model the attack according to the spoofing attack pattern (Fig. 2b). For simplicity reasons, we decided to model it directly in the ECU as a *OR_Event(5)*-gate with 5 event inlets. One of these inlets is the vulnerable one which can be used by the attacker to spoof a message of a flat tire. Therefore, we decided to model two attack steps (Fig. 4b). The first step handles the receiving of the ID-package, with a success probability γ of 7.5 %. Its success event will trigger the next step, which model the transmission of the fake message by the connection with the system's spoofing vulnerability. The related timespan $\delta = 0\,s$ means that this step is only executed once after triggering its *Set* in-port. The sensors are modeled with an *AND_Event_State*-gate which triggers an event out-port if the connected defect state of the tire is active and the deterministic time interval of 60 s is elapsed. Furthermore, the sensors have a defect state for indicating a failure. A more detailed model (defect states of the lamp and ECU, explicit antenna models) has been omitted due to simplicity reasons. The undesired event, called top level event (TLE), occurs if a tire is defective and the lamp is not indicating this withing 60 s. This timespan is represented in the SEFT by a *Duration*-gate, which triggers an event if the state at its inlet is active longer than the related timespan t. All parameters used in the analysis are shown directly in the SEFT models of Fig. 4) attached to their corresponding events.

In Fig. 5 the quantitative result of the analysis is shown as a graph. The x-axes represent the attack-rate λ of the spoofing attack step. Here it is not necessary to find an exact value for the attack-rate or a threshold if the rate is given as a range because of the situation, that the safety analysis shows better results with a higher attack rate. The reason for that is that such a spoofing attack "overrides" a defective sensor. The analysis has shown that the modeled attack has no negative influence on the safety property of the system. In this case, the approach was able to prove that there is no interference between safety and security. Thus, no countermeasures have to be taken to protect the system w. r. t. the system's safety. Nevertheless, in terms of the security property it would be advisable to think about appropriate security mechanisms to protect the driver's privacy. Otherwise, additional security threats are imaginable, like tracking of a car as shown in [1].

The results would look different if the attacker could exploit a DoS vulnerability. But therefore, the attacker has to install a jamming transmitter at the victim's car or drive with such a transmitter permanently in the transmission range of it. We could not find any attack which justifies this effort.

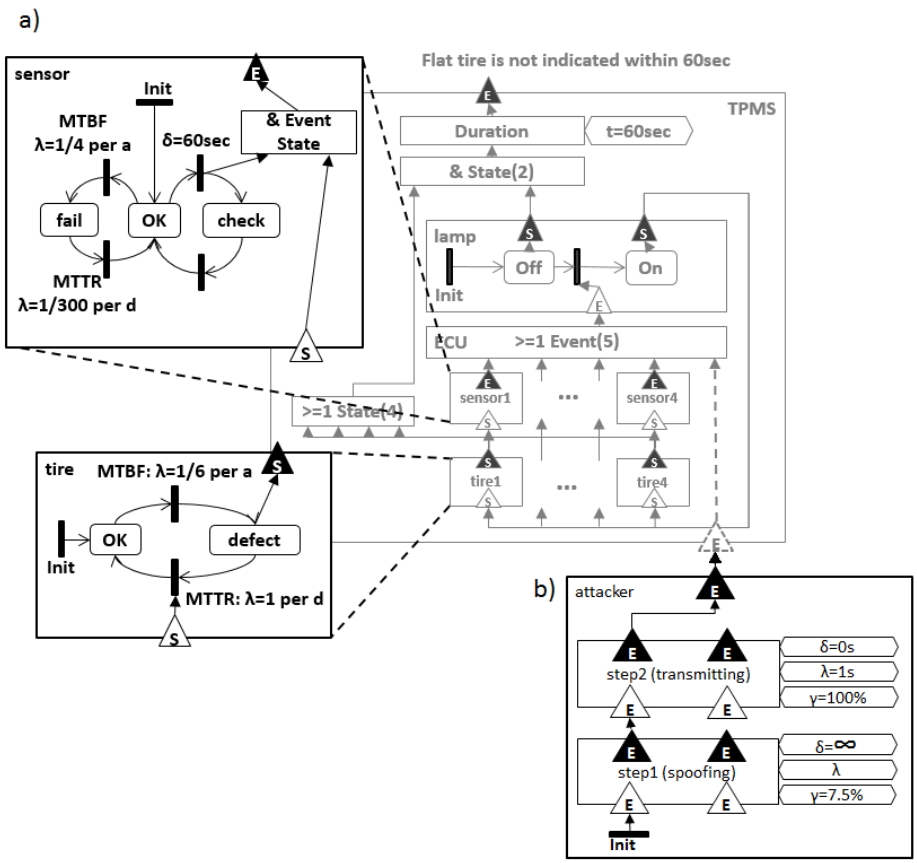

Fig. 4. a)SEFT of the TPMS b)SEFT of the attacker component

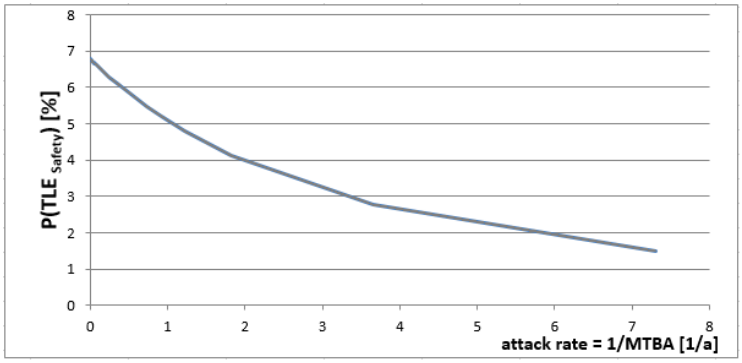

Fig. 5. Analysis results of the TPMS

7 Conclusion

In this paper a modeling and analysis approach was shown which is useful in common quantitative safety and security analysis. This technique allows the integration of security aspects into a safety model of an embedded system by using SEFTs. Therefore, we bring an explicit attacker model into account to model the behavior of an adversary. To deal with the security we made proposals of how attacks could be divided into smaller attack steps to make them manageable and analyzable. This prevents practitioners of getting stucked in an over-detailed model. To apply the aforementioned approach it is necessary to have quantified values on hand, e. g. the attack probability for specific attacks, which are not always available. Nevertheless, the decomposition of the attack quantities into smaller increments (time spans, rates, ...) and the possibility to analyze value ranges, as shown in this paper, can be helpful in solving this problem.

References

1. I. Rouf, R. Miller, H Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, I. Seskar: Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study. In: Proceedings of the 19th USENIX conference on Security, Washington, USA, 2010.
2. B. Schneier: Attack trees. Dr. Dobb's Journal, 1999.
3. A. Zimmermann, R. German, J. Freiheit, G. Hommel: TimeNET 3.0 Tool Description. Int. Conf. on Petri Nets and Performance Models, Spanien, 1999.
4. Embedded systems safety and reliability analyser. `http://www.essarel.de`.
5. H. R. Watson: Launch control safety study. Bell Labs, 1961.
6. B. Kaiser, C. Gramlich, M. Förster: State/event fault trees - A safety analysis model for software-controlled systems. In: Proceedings of the 23rd Int. Conference on Computer Safety, Reliability, and Security, Germany, 2004.
7. L. Piètre-Cambacédès, M Bouissou: Modeling safety and security interdependencies with BDMP (Boolean logic Driven Markov Processes). In: Proceedings of the Int. Conference on Systems, Man, and Cybernetics, Istanbul, Turkey, 2010
8. M. Steiner, P. Keller, P. Liggesmeyer: Modeling the Efects of Software on Safety and Reliability in Complex Embedded Systems. Computer Safety, Reliability and Security (SafeComp Workshops), Magdeburg, Germany, 2012.
9. P. J. Brook, R. F. Paige: Fault Trees for Securiy System Design and Analysis. Computer and Security, 2003.
10. Z. Benenson, E. O. Blaß, F. Freiling: Attacker Models for Wireless Sensor Networks. IT - Information Technology: vol. 52, No. 6, 2010.
11. R. Vigo: The Cyber-Physical Attacker. Computer Safety, Reliability and Security, Magdeburg (SafeComp Workshops), Germany, 2012.
12. I. N. Fovino, M. Masera: Through the Description of Attacks: a Multidimensional View. In: Proceedings of the 25th Int. Conference on Computer Safety, Reliability, and Security, Gdansk, 2006.
13. E. LeMay, M. D. Ford, K. Keefe, W. H. Sanders, C. Muehrcke: Model-based Security Metrics Using ADversary VIew Security Evaluation (ADVISE). In: Proceedings of the Int. Conference on Quantitative Evaluation of Systems, 2011.