Mälardalen University
School of Innovation Design and Engineering
Västerås, Sweden

Thesis for the Degree of Master of Science in Engineering - Software Engineering 15.0 credits

CASE STUDIES ON MODELING SECURITY IMPLICATIONS ON SAFETY

Aleksandar Matović
amc18003@student.mdh.se

Examiner: Marjan Sirjani
Mälardalen University, Västerås, Sweden

Supervisors: Aida Čaušević
Mälardalen University, Västerås, Sweden
Elena Lisova
Mälardalen University, Västerås, Sweden

September 24, 2019

Abstract

Security is widely recognized as an important property that is tightly interdependent with safety in safety-critical systems. The goal of this thesis is to conduct case studies on the implications that security attacks may have on the safety of these systems. In these case studies, we formally model the design of a robot arm system, verify its security against some potential attack scenarios, propose mitigation techniques and analyze their effectiveness. In order to achieve a thorough knowledge about the current formal verification approaches and select a proper modeling language/tool, we conducted an extensive literature review. We performed this review following a well-known approach proposed by Barbara Kitchenham. The procedure and outcomes of this review are detailed in this thesis. Based on the literature review, we chose TRebeca, (a timed extension of Rebeca), as the formal language to model the robot arm system, attack scenarios and mitigation techniques. Rebeca is an actor-based modeling language with a Java-like syntax that is effectively used to model concurrent and distributed systems. This language is supported by a full-featured IDE called Afra, which facilitates the development of (T)Rebeca models and verification of correctness properties (such as safety and security) on them. Among several functions provided by a robot arm system, we chose two important functions i.e., Stand Still Supervision and Control Error Supervision, which we believe would be interesting for attackers trying to get control over robot movements. In particular, attackers may maliciously manipulate the parameter values of these functions, which may lead to safety issues. In order to find suitable attack scenarios on these functions, we studied the most important security protocols used in safety-critical industrial control systems. We observed that these systems are vulnerable to several attacks, and man-in-the-middle attack is among the most successful attacks on these systems. Based on this study, we devised two attack scenarios for each function and modeled them with TRebeca. To mitigate these attacks, we proposed a redundancy technique, whose effectiveness was also assured by Afra.

Table of Contents

1 Introduction **4**

2 Work background **6**
2.1 System safety 6
2.2 Security implications 6
2.2.1 Threats and vurnerabilities 7
2.2.2 Integrating attack modeling with safety process 7
2.2.3 Risks and hazards 7
2.3 Actor-based modeling 8

3 Problem formulation **8**
3.1 Methods 9

4 Literature survey on formal modeling languages and their applicability for security **10**
4.1 Research method 11
4.2 The selection criteria 11
4.3 Results 11
4.4 UPPAAL 12
4.4.1 Overview 12
4.4.2 Aplicability for modeling security implications 12
4.5 Spin including verification language Promela 14
4.5.1 Overview 14
4.5.2 Aplicability for modeling security implications 14
4.6 Prism 16
4.6.1 Overview 16
4.6.2 Aplicability for modeling security implications 17
4.7 Rebeca 18
4.7.1 Overview 18
4.7.2 Aplicability for modeling security implications 18
4.8 SMV 19
4.8.1 Overview 19
4.8.2 Applicability for modeling security implications 19
4.9 NuSVM 19
4.9.1 Overview 19
4.9.2 Applicability for modeling security implications 20
4.10 FDR 21
4.10.1 Overview 21
4.10.2 Applicability for modeling security implications 21
4.11 Results and Analysis 21

5 Existing security analysis of frequently used protocols within the industrial control systems (ICS) **24**
5.1 Frequently used security protocols 24
5.1.1 Distributed Network Protocol protocol (DNP3) 24

5.1.2 Modbus . . . 25
5.1.3 Man in a middle attack . . . 26
5.1.4 PROFIsafe . . . 27

6 Description of chosen safety functions 29
6.1 Control Error Supervision function . . . 29
6.1.1 Function properties . . . 29
6.1.2 Execution logic . . . 30
6.2 Stand Still Supervision function . . . 31
6.2.1 Function properties . . . 31
6.2.2 Execution logic . . . 32

7 Adding security dimension to safety functions 33
7.1 An attack scenarios for CES . . . 33
7.1.1 Attack Scenario 1: OSA manipulation . . . 33
7.1.2 Attack Scenario 2: Cyclic Brake Check manipulation . . . 34
7.2 An attack scenarios for SSS . . . 35
7.2.1 Attack Scenario 1: Operating mode manipulation . . . 35
7.2.2 Attack Scenario 2: Position manipulation . . . 35
7.3 Possible strategies for mitigation and prevention . . . 37
7.3.1 Control Error Supervision proposed changes . . . 37
7.3.2 Stand Still Supervision proposed changes . . . 38

8 Discussion 39

9 Conclusions 41

References 49

1 Introduction

For safety-critical applications safety is the highest overall goal. From the 1990's it has been clear that the security and safety have to be more tightly connected to ensure overall acceptable system safety level. Brewer et al. in 1993. [1] realized how many similarities exist between the final goals of these areas. Accordingly, in the 1999. Eames et al. [2] proposed a technique for the integration of these previously separate methodologies. Not long ago safety-related systems have been closed stand-alone products. However, nowadays, those systems are able to be interconnected more frequently with implemented interfaces to the Internet to allow remote diagnostics [3]. Thus, security concerns become of high importance for assuring system safety, as well.

A documented example of how security breaches can directly jeopardize humans safety is the hacked insulin and drug infusion pumps [4]. US Food and Drugs Association issue a statement [5] resulting in regulation on how to address security breaches in a systematic way in order to secure the safety of medical devices [3]. Furthermore, there are some well-known cases of hacks which caused damage to the environment, including a case of a former employee who broke into a water treatment plant in Queensland, Australia and purposely spilled almost a million liters of raw sewage into local waterways [3]. If the security vulnerabilities have been exploited by a malicious attacker in the context of any industrial system, possibly fatal or severe injuries for nearby workers could occur. Hence, the running manufacturing processes could be jeopardized. That scenario would produce a counterproductive impact on company profit income. Besides, the biggest negative influence it would have on the systems manufacturer integrity.

There are existing security analyses on a system level such as vulnerability and threat analysis. Also safety and security co-analysis, e.g. FMVEA [6], SAHARA [7]. However, the progress in the research area has to be coupled with technologies being implemented in real life. Thus, it is necessary to connect those analyses with industrial case studies and a possible way to deal with such challenges may be the use of formal methods. With such an approach we are able to model safety and security functions and verify them against existing requirements. Formal verification approach adopted well defined mathematical logic to describe correct behavior of a particular system [8]. Due to the today system complexity, it is nearly impossible to conduct manual verification. Thus, by using the formal methods we hope to construct mathematically correct models of potential security attacks within our safety-critical system, in order to provide a detailed analysis in cases this would happen in the reality.

This thesis includes a survey on existing relevant model checking approaches and their applicability for security modeling. Moreover, a model checking tool Rebeca will be used to first model safety functions of an industrial robot, and second add security into such a model. A set of requirements will be verified against that model.

Furthermore, in this thesis we propose to model and verify several safety functions of a robot arm system using Rebeca tool. Rebeca (see Subsection 2.4) is an actor based modeling language including a formal foundation. Rebeca can be rec-

ognized as a reference model for concurrent computation [9]. We choose Rebeca actor-modeling language due to the its formal verification foundation but also the ability for modeling real-time application is taken into account. Moreover, the security analyses outcomes will be incorporated by modeling of security breaches and their implications on fulfillment of safety requirements.

The paper is organized as follows. In Section 2 a background necessary for understanding the concepts used in the thesis is presented. The background information concerning security implications, threats and vulnerabilities, attacks modeling into the safety process, safety implications, risk and hazards, actor-based modeling, and Rebeca-based modeling are presented in Problem formulation has been presented in Section 3 with the description of methods used in this thesis, and research questions. In the Section 4 literature survey on formal modeling languages and their applicability for security has been analyzed. In Section 5, existing security analysis of frequently used protocols within the industrial control systems (ICS) is elaborated. In Section 6 our description of the system safety properties is presented. In Section 7, our security aware approach is presented. In Section 8 research question from Section 3 has been revisited and answered. Lastly, the paper has been concluded with Section 9.

2 Work background

2.1 System safety

Safety critical systems are those systems whose failure could lead to consequences that are determined to be unacceptable such as loss of life, severe damage to the people, environment, etc. [10]. The overall goal of safety in this context is to ensure that a particular system does not produce any threats to the environment and its physical components. It is important to emphasize that is not possible to achieve an absolute risk-free state. Therefore, it is essential to set acceptable risk level in order to provide the appropriate protection to the system itself. One of the well-known cases of safety-critical system failure is the Mars Polar Lander [11] spacecraft which is lost in deep space because of the early termination of the engine igniting before the lander touching the surface. Moreover, Mars Climate Orbiter was launched to study the Martian climate and atmosphere. After misunderstanding due to the different measurement unit system used by engineers, the spacecraft encountered Mars on an orbit that brought it too close to the planet. Resulting in satellite being either destroyed in the atmosphere or it was re-entered planetary space after leaving Mars' atmosphere [12]. Many more examples of the safety-critical systems failures could be found in the book by Neumann [13].

Developing a safety-critical system with corresponding safety standard can often be a costly and time-consuming but required process. In numerous cases, the development of the safety case requires more funds than the development of the system itself [14]. Nonetheless, the safety process is necessary in most cases. Therefore, system requirements have to be linked with relevant certification.

2.2 Security implications

Security can be defined as a *"condition that results from the establishing and maintenance of protective measures that enables an enterprise to preform its mission of critical functions despite risk posed by threats to its use of information system"* [15]. Increasing use of consumer electronics, networks and their complexity in general, makes the process of ensuring the security more difficult. Therefore, it is necessary to provide the right layers of protection for safety-critical systems such as robot arm system. Accordingly, security evaluation has become an important requirement in design and management of computer networks.

Evaluating network security in a well-organized view can be done only if the hosts, network infrastructure, and relationship, as well as their interaction, have been taken into account [9]. In order for many useful services and functions to operate accurately, external communication must be permitted. Hence, this scenario can be potentially dangerous, as it opens up for an entire range of security implications that can be exploited by the malicious attacker [3].

2.2.1 Threats and vurnerabilities

A vulnerability can be defined as a "flow in the system that allows an adversary to realize a threat targeting one of the system assets. A concrete threat realization is an attack" [16]. Exploitation of system vulnerabilities would unquestionably cause security breaches and further implication on system safety. To produce a safe system by today's standards, security needs to be handled in a systematic way. One example of possible security implication on a safety-critical system originates from 2015 when security researchers Charlie Miller and Chris Valasek managed to take over Jeep Cherokee 2014 over the internet, which leads to Chrysler announcement that 1.4 million vehicles are going to be recalled for security updates before they can be safely run on the street [17].

2.2.2 Integrating attack modeling with safety process

Attack modeling presents a system from adversarial perspective [18]. Therefore, by designing this model we can observe system behavior and address possible vulnerabilities that can be exploited by attacker. The idea of integrating attack modeling process into the safety process is to show how security can endanger safety aspects of the system [16]. Furthermore, by including attack model we designing attacker "mindset" in order to discover its desired targets and then to provide necessary countermeasures and action in order to mitigate or fully prevent that attack.

In order to satisfy safety goal of the system, attack modeling process as an input uses a set of derived artifact including system definition [16]. There has been developed on a targeting system level of abstraction, e.g., vulnerability and threat analysis, and even safety and security co-analysis, e.g., FMVEA [6], SAHARA [7]. However, these approaches are applicable in the e.g., automotive domain, while the industrial robotic remaining outside the scope of this analysis.

2.2.3 Risks and hazards

A potential source of danger or possible threat can be defined as a hazard. In general meaning, every hazard in a safety-critical system can be described as a "state or a set of unsafe conditions such that when in a contact with environment conditions can lead to an accident - an unplanned event resulting in harm" [16]. One way to identify hazard and preform its analysis is by conducting a Fault Tree Analysis (FTA) of the potential causes of the hazard.

Fault tree analysis is tool which can be applied for analyzing and evaluating failure paths. For each potential hazard and potential hazard cause which could be the result of software failures, the analyst must extend the fault trees through the system hardware and software until a sensible set of software input and output variable values is identified. The value set associated with each hazard cause is then identified as a software hazard [19]. For the safety-critical system it is essential to identify potential hazards. Afterwards, these hazards are quantified according the

severity of potential harm (S), probability of exposure (E), and controllability of the resulting hazardous event (C)[7].

2.3 Actor-based modeling

The Actor model was originally presented by Hewitt [20] as an agent-based language. This model has been further developed by Agha into a concurrent object-based model [21]. The actor model is a mathematical model of concurrency computation based on the **actors**. Actors are computational agents which carry out their actions in response to incoming communications. Actors encapsulate procedural and declarative information into a single entity. Primitive actor languages are based on a pure message-passing semantics [22]. Actors are central and only objects in the model. This concept reminds us of the object-oriented programming. However, actors are unable to pass a modifiable reference to a different actor nor to modify a state of another actor. Thus, actors can only send messages, which is their primary purpose in this model.

The idea behind actor languages is to provide the syntactic constructs that can help the programmer from having to worry about the details of a concurrent execution [22]. Furthermore, advance of using the Actor model is asynchronous communication for passing messages and this process is enabled without any restrictions on messages arrival order. Having that in mind, fundamental concepts of the actor model are:

- Sending messages to another actors;
- Creating (finite) number of new actors;
- Asynchronous communication.

3 Problem formulation

Security issues in the safety critical system are fast evolving subject, as it is evident that possible problems with security in a system might jeopardize safety [3]. Hansen shows [23] that security and safety cannot be divided into separate fields anymore. We envision that potential of modeling security implication will give us an insight into a possible malicious attacker mindset and more importantly, this processes will serve us as a procedure for discovering potential security breaches and safety implications. One example of security exploitation which leads to safety-critical implications has been documented in 1999 when a teenager Johnathon James hacked into Marshall Space Flight Center in Huntsville, Alabama, and downloaded proprietary software for the International Space Station. The software supported the International Space Station's physical environment and was responsible for critical control of humidity and temperature for living in space [24]. Given that the today's systems rely on the communication protocols, both within the system and with the surrounding systems, the effect of security breaches is more visible. Thus, it is of the highest importance to be able to address such an issues as early as possible during the system development, as well as together with safety work, as there

might be direct impact on safety. Therefore, with such a stake in question, it is clear that modeling security implication is essential for ensuring that a system security breaches are as few as possible for the given situation.

The research question this thesis aims at addressing are:

- **RQ-1:** What existing approaches are addressing security by means of model-checking techniques?
- **RQ-2:** How to model security by using model checking approach in safety critical applications?
- **RQ-2a:** What are the possible implications on system safety that can be indentified by modeling security?

3.1 Methods

To answer the research questions the following methods will be applied:

1. Literature survey - the aim to collect information from the existing academic publications and learn about the level of use of model checking techniques while modeling and analyzing security.
2. Modeling security implications - predefined system specification and model it in a formal language. On that model the idea is to check already existing set of safety requirements. The use-case for modelling is a robot arm system. In the next phase for the same model we tend to enable security attacks and then check the existing requirements. Therefore, in order to check possible security breaches and vulnerabilities of observed robot arm system, we tend to use Rebeca language to verify if the necessary system requirements are satisfied. Thus, our aim is to check if the system can be manipulated and exploited by providing unspecified or faulty value which could eventually lead to the safety implications.

4 Literature survey on formal modeling languages and their applicability for security

This section presents a brief overview of the 43 papers that are considered relevant for answering the RQ-1. Papers are sorted in chronological order. Moreover, an overview of each considered model checking tool was presented in the introductory subsection and their applicability for modeling security was later taken into account.

The aim of this literature survey is to systematically describe what is the main concept behind model checking tools and their applicability for modeling security-related vulnerability analysis of the particular system. By conducting this research we hope to show which tool is more suitable and reliable for particular security interactions (e.g. protocols modeling or vulnerability analysis) and how the obtained results are related to our system. Furthermore, we use these models to answer the following question: How reliable is the code? Usually, the code comes with a set of requirements that must be satisfied. One way of checking if the requirements fold on the code is to it provide it with test cases and check if the solution meets the requirements. When the size of code and parallel components are large, doing a manual verification is nearly impossible. Hence, in such cases, we use model checking tools.

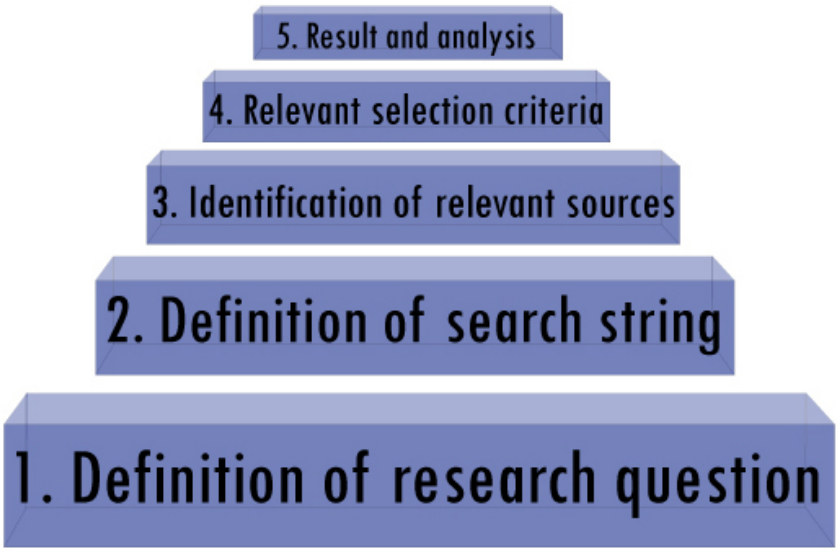

Figure 1: Phases of literature survey

4.1 Research method

For conducting this literature survey we have been inspired by B. Kitchenham approach [25]. Therefore, we divided literature survey into 5 phases presented in Figure 1 as our research methodology. Moreover, in Phase 1, we identify RQ-1 as relevant for conducting a literature survey. In phase 2 we identify the following boolean:

(***"model-checking"*** AND ***"security"*** AND ***"modeling"***)

In the phase 3 we applied this research string into five major databases: SpringerLink[1], ScienceDirect[2], IEEE Xplore digital library[3], Web of Science[4],ACM digital library[5]. In order to collect and analyze the data required for this investigation we used the Mendeley application.

4.2 The selection criteria

After we applied the search string to the specified databases, as a result, we had 18892 papers in total. In order to obtain highly relevant papers concerning our research, we first identify the following criterion: *"The publication must contain model-checking and security-related issues"*. After initial results, we eliminated irrelevant ones by reading titles of each paper. Furthermore, after this step, we had 969 papers. In order to lower the number of papers that are not related to our research, we applied the inclusion criterion to the abstracts and introductions of the papers. Thus, we had 140 papers. In the last phase, each paper was read and the inclusion criterion was applied to the full text. As a result, we got 43 highly relevant papers which are briefly described in the following subsections.

4.3 Results

Source	Phase 1	Phase 2	Phase 3	Phase 4
IEEE Explore	527	97	38	19
ACM digital library	10247	562	26	6
SpringerLink	6145	181	51	11
Web of science	116	17	10	2
Science Direct	1857	112	15	5

Table 1: Sources

[1] https://link.springer.com/
[2] https://www.sciencedirect.com/
[3] https://ieeexplore.ieee.org/
[4] https://webofknowledge.com/
[5] https://dl.acm.org/

4.4 UPPAAL

4.4.1 Overview

Larsen et al. (1997) [26] presents UPPAAL structure and main features of this toolbox collectively developed by Uppsala University and Aalborg University. This paper gives us an extensive overview of each main model functions and design criteria of this tool. Authors further describe UPPAAL's primary elements i.e. simulator, model checker, and a description language including a graphical representation of UPPAAL model structure. According to the authors UPPAAL model checker has been used as modeling or design language in order to represent an associated network of timed automata including data variables extension. For a better understanding of UPPAAL model structure observe the following model presented in Figure 2. In this example model consists of two components i.e. A and B including control nodes A0, A1, A2 , A3 and B0, B1, B2, B3 respectively. On the right side of Figure 2, the declaration of used variables can be seen. Furthermore, the model uses two clocks x and y, one integer n and one channel a. The edges of the shown model automata have three types of labels:

- A synchronization action - is performed when the edge is taken and processes can be synchronized over the common channel;
- A guard expression - a boolean expression that has to evaluate to true for an edge to be enabled.
- The update expression - the action containing that update is activated.

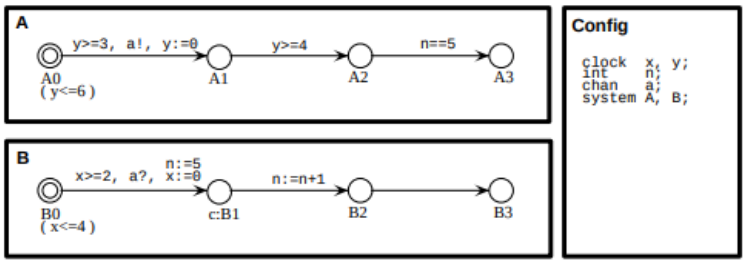

Figure 2: An example of UPPAAL model structure [26]

4.4.2 Aplicability for modeling security implications

Corin et al. (2004) [27] analyze the well-known Needham Schroeder protocol. Authors proposed a method for security protocol which takes timing issues (e.g. timeouts and retransmissions) into account. Uppall is used as the modeling tool of choice to simulate security protocols against established safety goals (e.g. authentication) in a real-time scenario. Furthermore, authors proposed a model to improve before mentioned protocol and present a subtle attack based only on the timing aspect.

Dıaz et al. (2004) [28] analyze E-commerce interaction between client and server sides. Hence, authors chose the well-known Transport Layer Security (TLS)

protocol and later present a way to ensure the properties of this protocol. Hence, in this paper authors managed to model TLS handshake actions, then to validate and verify its properties by using the UPPAAL model checker. To ensure the protocol desired properties authors used the following methodology: model the system, validating the correctness, specify the property and verify the system.

Mondal et al. (2009) [29] proposes a formal technique using UPPAAL in order to execute security analysis. Authors further described the safety and liveness properties of the system. Furthermore, according to the author's Role Based Access Control (RBAC) lack such before-mentioned analysis. Therefore, RBAC extension is proposed. Authors also proposed a state transition model for GTRBAC which is a well known RBAC model capable of mapping properties and component behaviors such as user, roles and permissions.

Saghar et al. (2015) [30] by using UPPAAL for modeling RAEED (Robust formally Analysed protocol for wirEless sEnsor networks Deployment) protocol authors succeed to ensure safety against attack called Hello Flood which is well-known Denial of Service (DoS) attack. Authors point out that formal verification methods perform excellently in presence of Hello Flood attacker. Hence, they came to the conclusion that UPPAAL confirmed that RAEED is protected from the hello flood attack in all 1024 performed symmetric link topologies.

Kassmi et al. (2016) [31] analyze issues regarding formalization and verification of web service in order to meet specified security requirements. A Health Care case study is analyzed for the purpose of this research. This paper is based on the previous work from the same authors about DIVISE Framework (Discovery and Visual Interactive web Service Engine) used to formalize web service requirements. Since the non-functional requirements are not handled properly by DIVISE Framework, authors propose specified security requirements model using timed automata.

Amin et al. (2017) [32] proposes a model in order to mitigate Black Hole attack from ARRIVE protocol in a wireless sensor network. As a result, the model has shown that UPPAAL is able to eliminate bugs and successfully verify system properties. Authors suggest that this approach can be useful if it is applied in the early stages of development in order to eliminate any chances of later implications.

Sun et al. (2015)[33] analyzed EPC (Evolved Packet Core) networks and their security issues. Furthermore, this network is used in several fields of essential importance. Hence, authors describe network properties, then the classification of the existing security attacks are presented. Finally, the correctness of the proposed method is checked using the UPPAAL model checker.

Gadyatskaya et al. (2016)[34] proposed a method for partitioning the modeling of the attacker's and defender's behavior from a particular attack-defense tree, attack tree represent an extension of traditional attack tree. Authors used Uppal versions SMC and Stratego in order to model the environment with relevant activities. Furthermore, a framework for analyzing interactions between attackers and defenders is presented in this paper.

4.5 Spin including verification language Promela

4.5.1 Overview

Holzmann (1997) [35] SPIN represent popular model checking tool developed by Unix group of the Research Science Center at Bell Labs. Acronym SPIN means "Simple Promela Interpreter". Promela (**Pro**cess **M**eta **La**nguage) is design in order to describe and verify systems. SPIN actually do not perform model checking immediately, but first creates C programs for another suitable model checker. According to the authors, this technique causes less memory consumption and improves performances. SPIN is similar to Rebeca due to its possibility to model asynchronous distributed algorithms. Properties can be verified by using Linear Temporal Logic (LTL), these formulas are later translated and converted into Büchi automata which is part of the SPIN algorithm used for model checking.

4.5.2 Aplicability for modeling security implications

Josang (1995) [36] investigated whether SPIN as a formal-method can be used to check usually undetected (by traditional methods) protocol vulnerabilities. The author points out that even well-known protocols have flaws and weaknesses. According to the author, the well-known tool concerning security protocol verification is BAN logic and its summary is also presented in this paper. However, the manipulation of intended protocol design can be spotted by the SPIN which is not the case with BAN logic. A. Josang further investigates the X.509 [37] protocol which is basically a digital certificate that uses X.509 public key infrastructure (PKI). After practical realization and estimating the state space author concludes that most types of flaws present in the protocol can be found, albeit not all. Hence, author propose a way of dealing with FSM (finite state machine) verification of security protocols.

Maggi et al. (2002) [38] investigate the use of SPIN in the verification of security protocols. Furthermore, these protocols according to the authors appeared to be error-prone to the potential attacks. However, with formal verification methods, this issue can be handled properly. Authors use well-known Needham-Schroeder Public Key Authentication Protocol adopted for modeling purposes during this research. Work is separated into two segments i.e. protocol properties description, and description of attacker behavior. As a result, the method to deal with the construction of the intruder model using a case study is proposed as well.

Singh et al. (2003) [39] present a technique for detecting specific malicious behavior in worms. During first work instance code is decomposed in the forms of predicates which represents program properties. To classifies, worms and virus behavior authors used the term "organ" as an illustration of a functional organ. The program was translated to a finite model and later analyzed by the SPIN model checker to verify executables against the presence of malicious programs. As a result, authors introduced a method for semantic capture the presence of malicious behavior and to later verify that properties using SPIN.

Nakajima (2004) [40] present an approach for verifying safety and security aspects. Furthermore, as the main aim author propose one framework for handling

these two aspects which would be later analyzed and verified by using SPIN model checker. Author analyzed web service framework, particularly BPEL (Acronym for Business Process Execution Language for Web Service) which is standard description language of control service flows build upon WSDL (Web Service Description Language). The lattice of appropriate security levels are also defined and discussed (i.e. top-to-bottom: Top secret, Secret, Confidential, Unclassified)

Xiao et al. (2006) [41] proposes a method for constructing the PROMELA model of the cryptographic protocol which is later used for modeling Helsinki protocol specified by the TLA (Temporal Logic of Action), model is verified by the SPIN model checker. As a result, an output chart of the attack has been found and presented. Author points out that modeling of the intruder capability represents a highly challenging task due to its complexity. Moreover, according to the authors, this paper can be helpful for verifying properties of cryptographic protocols and examination of its flaws, which can be later improved by applying formal methods.

Jiang (2008)[42] proposes new trace semantics for verifying security properties of cryptographic protocols by using SPIN model checker for verification. To demonstrate the proposed technique author by modeling two attacks on the TMN (Telecommunications Management Network) protocol. Furthermore, the author emphasized that due to the obtained result and versatility of this method other network-related security protocols can used this method for verification.

Wang et al. (2008)[43] used SPIN for detecting possible vurnerabilitates in the AACS drive-host (distribution system for recordable and pre recorded media) and also proposed an version of authentication protocol based on the Dolev-Yao attacker model [44].

Li et al. (2009) [45] analyze SET (Secure Electronics Transactions) protocol. To ensure one of its main properties author used SPIN in order to verify purchasing processes. An abstract model is specified using PROMELA. In the experiment, authors fill the model with the malicious data which could be obtained by the malicious attacker. By using the SPIN author are being able to discover potential flaws in the model.

Ma et al. (2010) [46] proposed a method in order to verify system security information flow and according to the authors the system should be verified and validated in three areas: checking system requirements, consistency, and completeness of security policies, validation of security solution due to the confirmation for the security policies. Furthermore, in order to verify these aspects, the authors used SPIN model checker.

Basagiannis et al. (2010) [47] analyze Needham–Schroeder security protocol. According to the authors attempt to enumerate messages often lead to the state space explosion. Thus, proposed Message Inspection (MI) method analyzed exchanged messages metadata. SPIN model checker was used in order to analyze the data, and the experimental result is presented.

Chen et al. (2011)[48] analyzed Wireless Sensor Network (WSN) environment by applying security by applying formal verification methods. Networks are analyzed by the SPIN model checker, and according to the authors this model checker proves to be useful and provides feasibility for setting new idea for ensuring security system properties.

Basagiannis et al. (2011) [49] proposed intruder model as an attempt to detect vulnerabilities in security protocols called Micromint and PayWord. In order to analyze the correctness of the authors of the given protocol used the assertions of invalid end states supplied by the users.

Chang et al. (2012) [50] analyze security properties of lightweight response mechanism (LWRM) by proposing the PROMELA model. Furthermore, based on the found error trace authors designed an attack model using SPIN model checker. Finally, the authors verify properties and propose a model in order to mitigate found challenges.

Kushik et al. (2012) [51] analyze software made by students used for array processing algorithm application. Furthermore, in order to detect vulnerabilities in C program authors used SPIN model checker. Authors point out that due to the existing disadvantages regarding static code analysis it was necessary to develop and use dynamic detection of vulnerabilities.

Chen et al. (2016) [52] propose a method to ensure the properties of security protocols. Furthermore, the authors emphasize that SPIN has good model checking abilities and LTL (Linear temporal logic) support. As a result, the mentioned method is used in order to check the authentication of the security protocol. According to the authors, this method is proven to be effective and efficient.

Madhusudhanan et al. (2017) [53] analyze smart grid which is a system with multiple connected smart devices, the security of such a system is of high importance. Authors analyze the OS of the smart grid by using Access Control Mechanism (ACM). Therefore, in order to verify security properties, authors used SPIN model checker that takes TS and SPIN as an input with an aim to enhance security within smart grid networks.

4.6 Prism

4.6.1 Overview

Kwiatkowska et al. (2002) [54] presents PRISM fundamentals principles. This open source tool is developed at the University of Birmingham for the purpose of analyzing probabilistic systems. Probabilistic model checking provides a way to model and analyze likelihood of the occurrence during the execution of a particular system. It is important to emphasize what is the difference between formal and probabilistic verification. Probabilistic model checking is a formal verification technique used for modeling and analyzing systems that exhibit probabilistic behavior. PRISM model checker can be applied to ensure correctness of quantitative analysis (e.g. failure rate or system performance). For constructing initial model PRISM is required to use BDDs (Binary Decision Diagrams) and MTBDDs (multi-terminal BDDS). In Figure 3 overview of PRISM, components are presented. Furthermore, a high-level model is described using PRISM language description, on the second phase transition from high-level language and creating a set of reachable states are executed by using a matrix or graph-based approach. Model is generated into DTMC (Discrete-time Markov chain), MDP (Markov Decision Process) or CTMC (Continuous-time Markov chain) and together with the variation of logic formulas (e.g. PTLC/CSL/LTL) provided to the model checking process. Finally, depending

on the given inputs PRISM can produce quantitative analysis and counterexamples.

PRISM is used in the areas of wireless communication protocols and quantum cryptography. Besides, nowadays avionics and multimedia systems also find this model checking tool as relevant.

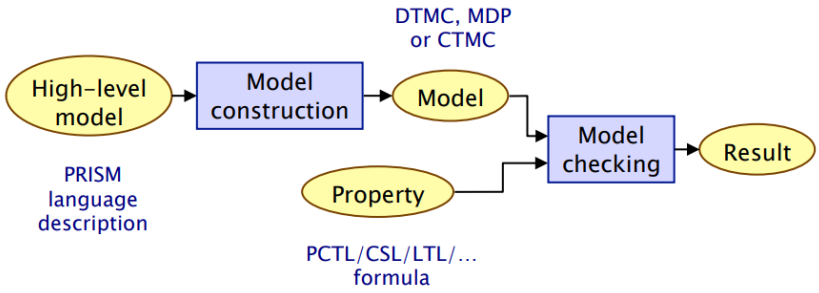

Figure 3: PRISM overview [55]

4.6.2 Aplicability for modeling security implications

Steel (2006) [56] designs a formal analysis framework for incorporating an attack with the aim to take advantage of PIN (Personal Identification Number) information. This information is crucial for encryption and sensitive information transfer in banking networks. Therefore, the author used PRISM probabilistic model checker to construct and find the most effective attack.

Basagiannis et al. (2008) [57] introduce a probabilistic formal analysis framework for computing DoS (Denial of Service) potential security attacks. Furthermore, the author's aim was to provide a comparative analysis of implementation with alternative choices. They also developed a model for HIP (Host Identity Protocol) which is applied as a protective mechanism against Denial of Service attacks.

Deshpande et al. (2011) [58] present a DNS BAA (Bandwidth Amplification Attack - distributed DoS attack). Authors used PRISM to construct continuous Time Markov Chain and to propose three counter measures regarding this attack. Furthermore, a cost-benefit analysis is also presented. Results show that DNNSec used for ensuring security acceptance level against DNS attacks is more vulnerable to a BAA compare to the DNS.

Zonghao et al. (2016) [59] focus on harmful network propagations by providing protection strategies. Based on the model of e-mail virus propagation authors provides three protective measures (i.e. free propagation installation and periodical protection). In order to verify proposed strategies authors used PRISM as a tool of choice. When the problem scale increased approximate probabilistic method is applied.

Huang et al. (2016) [60] describe protocols for ensuring the security of key exchange interaction. Authors used PRISM to test security properties of this interaction. According to the obtained result from this analysis, significant difficulties of quantum security protocol and analysis have been found.

Alexiou (2016) [61] analyze NFC (Near Field Communication) mostly applicable to the payment or ticketing services. In order to provide an accurate data

transfer, two devices need to be in close range (not more than 10 cm). According to the authors, this technology can be exploited by the malicious attacker which could obtain sensitive user information. Authors user PRISM to check NFC parameters and by combining attacker behavior authors are able to provide a model of relay attack. As a result, authors discuss potential countermeasures and preventions.

Chen et al. (2016) [62] propose a quantitative analysis framework based on the Bayesian network in order to identify potential attack scenarios using probabilistic PRISM model checker. This analysis was conducted in two phases. This framework is later validated by appropriate case studies.

4.7 Rebeca

4.7.1 Overview

Rebeca (Acronym for Reactive Objects Language) [63, 64] is an actor-based modeling language and its implementation is based on the theories proposed by Agha [21] and Hewitt [20]. Rebeca is designed to model and verify concurrent and distributed systems with the goal to bridge the gap between software engineers and the formal methods community by being a usable and at the same time analyzable modeling language [65]. Rebeca syntax is Java-like and in this actor-based language shared variables between actors is not possible. Asynchronous message passing is one of the main features. Furthermore, there is no blocking (send or receive) statement. It is crucial to emphasize that Rebeca as an actor-based modeling language inherits all the above-mentioned concepts (see Subsection 2.3) regarding actor modeling.
To support modeling in different areas Rebeca introduced its whole family of extensions [65].

- Extended Rebeca - Globally asynchronous locally synchronous systems;
- RebecaSys - Hardware/software co-design;
- Variable Rebeca - Product lines of actors;
- Broadcasting Rebeca - Actors with broadcasting abilities;
- Wireless Rebeca - Used for Ad-hoc mobile networks;
- Timed Rebeca - Used for realtime actors;
- Probabilistic Timed Rebeca - Probabilistic real-time actors.

4.7.2 Aplicability for modeling security implications

Shahriari et al. (2010) [9] proposed the model with an aim to analyze TCP (Transmission Control Protocol). Furthermore, the authors used the formal part of Rebeca and available tools to apply this actor-based language to the model networks of hosts and provide counter-measures. Thus, by using Rebeca language author was able to construct a detailed model of hosts and to design three attacks scenarios.

In the first scenario, simple security properties are tested by preventing the user from connecting to the host by sending *Reset* packets. Similarly, in the second scenario attack is designed by sending *SYN* packets regardless of the established communication rules. In the third, more complex scenario, the multiphase attack was constructed in order to analyze the Mitnick attack. According to the authors, Rebeca can be used for checking the requirements of security protocols due to its asynchronous and event-based properties.

4.8 SMV

4.8.1 Overview

McMillan (1993) [66] in order to solve state explosion problem present symbolic model checking method. This system can verify programs by using temporal logic formulas. NuSVM model checker represent a version of SVM. Property that can be checked with SVM are: CTL, LTL, safety, deadlocks, fairness, and lifness.

4.8.2 Applicability for modeling security implications

Xia et al. (2009)[67] verify weaknesses in the design of WTP (Wireless Transition Protocol). Author introduced three steps for analysis. 1. Wireless Transition Protocol is modeled by FSA (Finite State Automata). 2. CTL (Computation Tree Logic) is used to specify protocol properties. 3. After the model is specified and modeled author used SMV to verify results. Furthermore, results show previously unreported existing security vulnerability (i.e. deadlock in WTP).

Ritchey et al. (2000)[68] analyze vulnerability in the networks of multiple hosts. Authors introduce a technique which can be helpful for modeling and analysis of network security by conducting this technique user is allowed to specified multiple attacks scenarios using the same model description. Authors provided a test case of multiple attacks scenarios by using the SMV tool, model checker was chosen due to the authors previous familiarity with this tool.

4.9 NuSVM

4.9.1 Overview

Cimatti et al. (1999) [69] present NuSVM as a new symbolic model checker developed by Carnegie Mellon University (CMU) and Istituto per la Ricerca Scientifica e Tecnolgica (IRST). NuSVM is designed as an extension of SMV (see Subsection 4.7). According to the authors, NuSVM is re-engineered and updated from three points of view. First, the GUI interface and interaction shell which allows LTL model checking is introduced. Second, the modularity of the system is much more visible and open. Third, allegedly maintenance and well-documented system with easier to modified source code is presented. According to the authors, these features greatly improves implementation quality. NuSVM is proven to be suitable for modeling hardware circuits. However, one model checker can be more appropriate depending on the given scenario.

4.9.2 Applicability for modeling security implications

Adyanthaya et al. (2010)[70] analyze Kerberos network protocol for possible weaknesses. Work is motivated by previous work on the Kerberos protocol concerning model checking tools. Therefore, the authors introduce a simple model with an incorporated replay attack. Moreover, a concept for overcoming this attack is presented as well.

Mundra et al. (2011) [71] design a model of Inter Realm Authentication protocol and users interaction in Kerberos protocol. In the distributed systems environment resource over network can be accessed only by authenticated users. Therefore, it essentially important to prove nodes identities over networks, According to the authors, NuSVM model checker is the best option for construct and verify such a model. Authors also demonstrate the attacker presence in the system and provide generated counterexample.

Jacob et al. (2013) [72] analyze a service used for sending sensitive information over the network. According to the authors common protocols for conducting this transfer is highly insecure. Therefore, the Zero Knowledge protocol provided necessary authentification with sending only mathematical computations of particular information. By using NuSVM authors presented a model for proving authentification properties of this protocol. As a result, authors have shown that the user is able to detect if the malicious attacker is trying to gain access to the system.

Safarkhanlou et al. (2015) [73] proposes a service model regarding secure system state maintenance which is used for an antivirus program. The behavior is specified by using machine diagrams and later translated by using CTL (Computer Tree Logic) language. After initial translation and specification model is verified by using NuSVM model checker. Furthermore, authors point out that obtain experiment results shown that all necessary requirement for the given model is satisfied. A. Safarkhanlou et al. also discovered some logical weaknesses such as reachability, fairness and deadlock free.

Yamaguchi et al. (2018)[74] proposes a method in order to analyze possible attack in IoT (Internet of Things) system. To model, the given IoT system authors used Petri Net PN2 which is agent-based modeling language. After initial system model specification author used NuSVM as a tool for analyzing potential attacks to the system. Furthermore, as an application model, IoT equipped factory is presented. Moreover, the model is later tested by implementing an attack known as DNS cache poisoning.

Shrestha (2018) [75] present a case study of ICS (Industrial Control System) where is necessary to mathematically prove system properties before its deployment. Author analyzes control module for temperature control. To translate the control module model into model appropriate for the NuSVM model checker author used manual verification. As a result, possible vulnerabilities for the given model is found and discussed in the paper.

4.10 FDR

4.10.1 Overview

Gibson-Robinson (2014) [76] present new model checking approach with an aim to transform high-level specification into low level execatuble specification. FDR can be describe as an model checker but more presice description is refinment checker. This tool is designed in order to check expression in the CSP (Communicating sequential processes)

4.10.2 Applicability for modeling security implications

Lowe (1996)[77] analyze Needham-Schroeder protocol and identified potential vulnerabilities by applying FDR checker. Furthermore, protocol and attacker behavior are formed in CSP (Communication Sequential Processes). FDR checker is later used to ensure and verify that there is no present attack on a small system. According to the author, this result proves that there is no existing attack on a more general system.

Lu et al. (1999)[78] analyze correctness properties of the Secure Electronic Transaction (SET) which is the protocol used for secure money transaction jointly developed by Visa and MasterCard. Authors verified options provided for this protocol (i.e. payment transactions, purchase request, and authorization of payment). By using FDR authors prove that SAT satisfied all before mentioned interactions.

Kraetzer (2010)[79] introduce a six-step procedure based on the given application scenario to describe, model and verify XML-structures watermarking application and communications network. Furthermore, author introduces a method for automated translation of the watermarking protocol to the CASPER notation.

4.11 Results and Analysis

This subsection presents an analysis of the conducted literature survey. Each analyzed paper contains only essential information about the scope of the research.

Moreover, the Spin model checker was used in 16 papers. We recognize that SPIN is most frequently adopted for checking protocols security properties as found in 8 papers. Other papers mostly investigated software vulnerabilities with the construction of models used to prevent potential attacks. Therefore, due to the SPIN popularity containing a high number of identified papers in this research we can conclude that reported results showing that SPIN is used the most for the purposes of security model checking. On the other hand, Rebeca is the least used with only one reported paper. However, we identify potential for security modeling in Rebeca due to its formal verification and closeness to the software development community by having java like syntax.

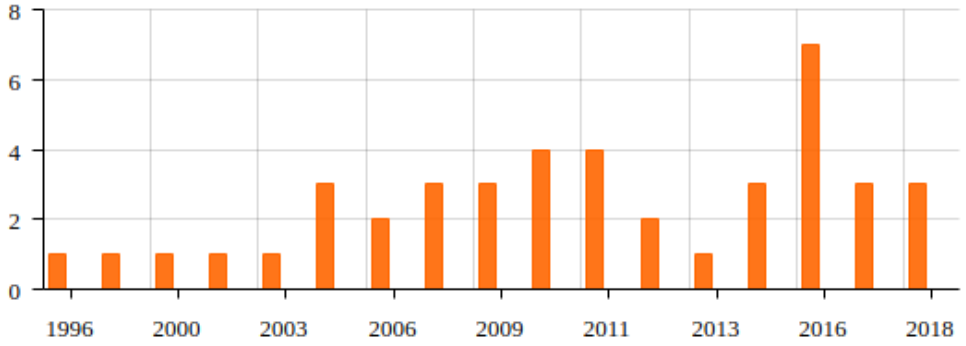

Figure 4: Paper distribution per year

In Figure 4 we can observe trend of publishing identified papers. Therefore, we can notice that from 1996. security modeling using model-checking tools is getting more and more popular.

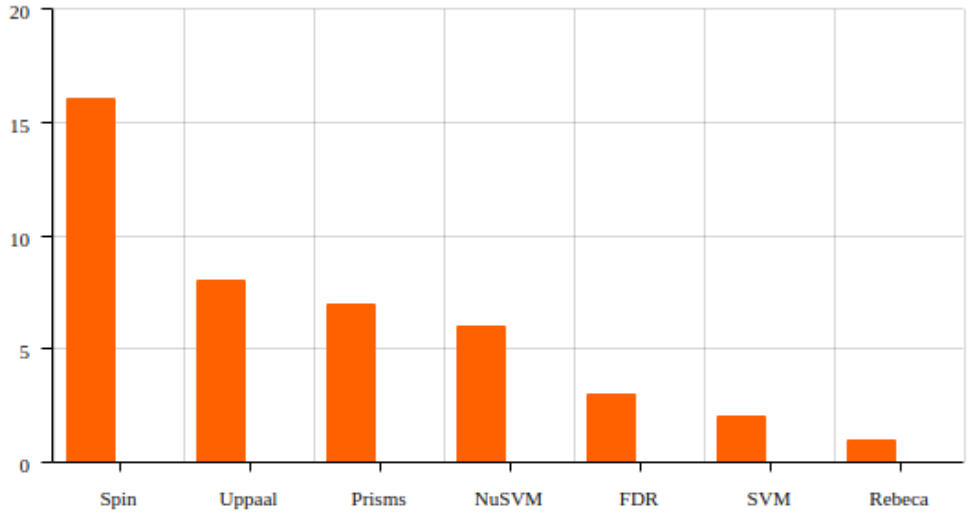

Figure 5: Paper distribution per model checker

In Figure 5, we see the distribution of use of all identified model checkers over the past years. Therefore, it is clear that Spin with 16 identified papers is the most popular, and Rebeca is the least one with only one identified paper.

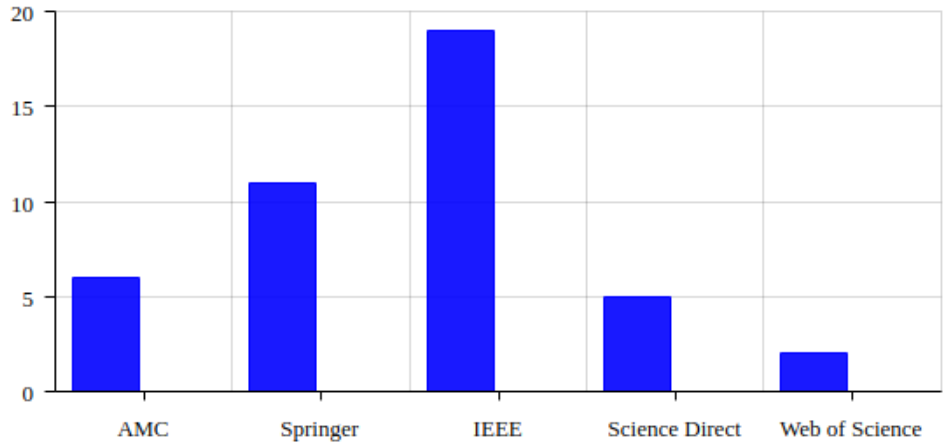

Figure 6: Paper distribution per source

We also identify relevant databases and paper distribution accordingly. Therefore, in Figure 6 we can observe that IEEE Xplore Digital Library has been the source of the most relevant papers in this study with 3.61%. Furthermore, ACM Digital Libary has the least relevant papers with 0.06%. SpringerLink has 0.18%, Web of Science 1.72% and Science Direct holds 0.27% of relevance.

.

5 Existing security analysis of frequently used protocols within the industrial control systems (ICS)

In this section, we tend to analyze the current security setup and publications within a distributed industrial control system (ICS). Security analysis of our case study in Section 8 as an assumption takes the fact that the system is already compromised and the attacker can then manipulate the values of functions without being detected. Therefore, to conduct full security analysis and show how malicious attacker can first exploit security protocols within ICS and gain access to the system properties and then manipulated the values of the system functions we are going to describe the most frequently used protocols in the ICS and its vulnerabilities by showing how this protocols can be potentially exploited.

It is from essential importance for every safety-critical system to posses the right layer of protection to ensure that real-time communication and availability requirements are satisfied. Therefore, security protocols are from essential importance when it comes to securing data integrity and transmissions over the network. According to Drias et al. [80] which extensively elaborated taxonomy of attacks present in ICS, there are multiple commonly used protocols. Guided by their analysis and results present in literature we are going to elaborate and analyzed potential security vulnerabilities of most popular security protocols.

5.1 Frequently used security protocols

5.1.1 Distributed Network Protocol protocol (DNP3)

Distributed Network Protocol (DNP3) is mostly used as a Supervisory Control And Data Acquisition (SCADA) protocol. It is mostly applied in control applications of electric utilities with more 75% of usage in the North America [81]. According to the [82] DNP3 protocol lacks the necessary security mechanism. Moreover, they presented and proposed a security framework for the DNP3 protocol. DNP3 is not fitting into OSI architecture of 7 layers, it's supporting much simpler mode of 3 layers which is proposed by International Electrotechnical Commission (IEC). According to the IEC proposed model, there is a lot of enhanced options called Enhanced Performance Architecture (EPA). However, DNP3 updates the proposed model by adding the fourth layer which enabling message segmentation (See Figure 9).

According to the CNN footage of the "Aurora" experiment conducted by Idaho National Laboratory scientist who managed to turn the electricity down in power generator [83]. However, this experiment is conducted in nearly perfect condition there is a lot of risks that similar kind can be performed in real-world conditions. East et al. [81] presented an extensive analysis of the DNP3 protocol, with 28 conducted attacks used to manipulate traffic and data within the network. They showed how these attacks can be used in real-time scenarios and proposed certain mitigation scenarios while emphasizing that many more attacks remain to be discovered in a near future.

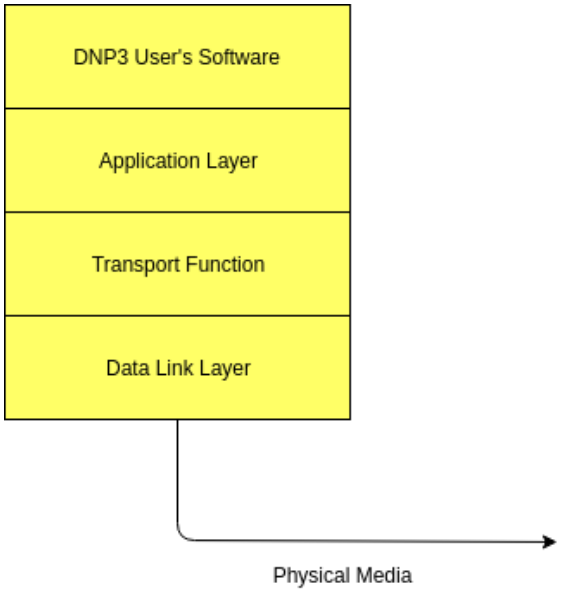

Figure 7: Reinforced OSI model for DNP3

5.1.2 Modbus

Modbus protocol represents typical Supervisory Control And Data Acquisition (SCADA) protocol, developed by Modicon in 1979. Nowadays, it is becoming a de facto standard in the automation industry. Modbus protocol operates completely independent of hardware structure [84]. There is a master/slave method present in this communication process. Moreover, the master device sends a necessary request to the slave device which response in with correct information. This protocol can be used in an embedded environment as well. To show that Modbus is reliable in this condition, Peng et al. [84] designed Modbus master services under Linux embedded system are proven to be stable and consistently good in performance.

Multiple attacks groups can be performed to Modbus protocol some of them are specifically targeted Modbus TCP and Serial protocols [85]. According to the Fovino et al. [86] there is several of Modbus vulnerabilities that can be exploited if malicious user tries to penetrate protocol security features. Therefore, we are going to focus on the following types of attacks:

- Modbus Denial-of-Service (DoS attacks); This well-known types of attack can be performed to prevent the system from functioning at the right level by sending messages from master to remote terminal unit RTU by using the legitimate channel of communication. This type of attack will disable the system, however, data within the system will be impacted. Therefore, this kind of attacks can be performed if an unauthorized user is trying to communicate with the system.

- Unauthorized Command Execution attack: One of the example how Modbus protocol can be exploited is presented by Carcano, et al. [87], where malware specifically designed to infect network and to make communication from the master device to slaves sent automatically;

- Man-in-the-Middle Attacks, this type of attack is specifically interesting in our security analysis, due to concept similarities presented in the next section, therefore this attack is extensively described in the following subsection;
- Replay Attacks: if the attacker can be used legitimate ways of communication and repeatedly send messages within the network to slave devices or from itself this could cause potentially serious consequences [86]

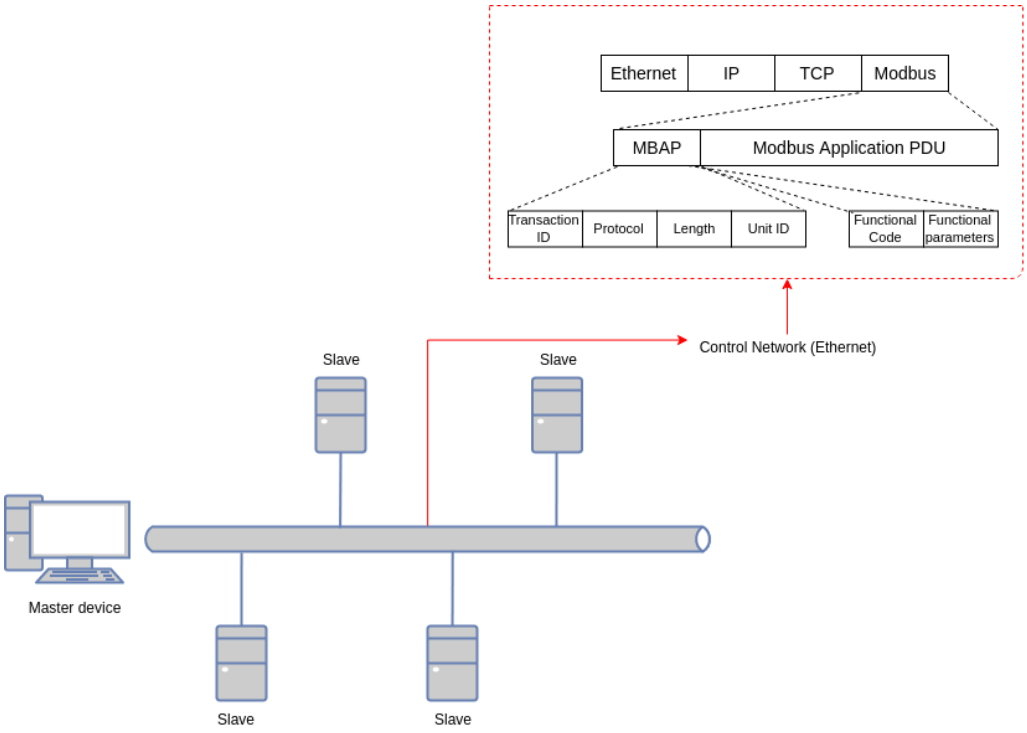

Figure 8: Modbus TCP arhitecture

5.1.3 Man in a middle attack

From our point of view, the Man in a middle attack can be defined as a potentially most complex and dangerous way to jeopardize the infected system. Due to its nature, it is especially hard to detect which communication unit or in this case slave device is infected and what damage is caused. On Figure 10 we can observe how unauthorized user which is impersonating itself as a slave device can gain access to the whole network. Therefore, an attacker can read messages, send messages automatically to the slave devices and eventually compromise the network of connected peers. Huitsing et al. [85] listed potentially consequences of fabricating the master device and data within the network.

There are known attempts of detection and prevention of this kind of attacks. Furthermore, Eigner et al. [88] were able to detect and prevent Man in a middle attack by using k-Nearest Neighbors algorithm with Bregman divergence to decide what can be considered normal behavior. However, according to the authors, this

research and findings are conducted in the ideal condiction within the lab environment. Therefore, it is questionable when this kind of attack prevention can be applied in the real industry.

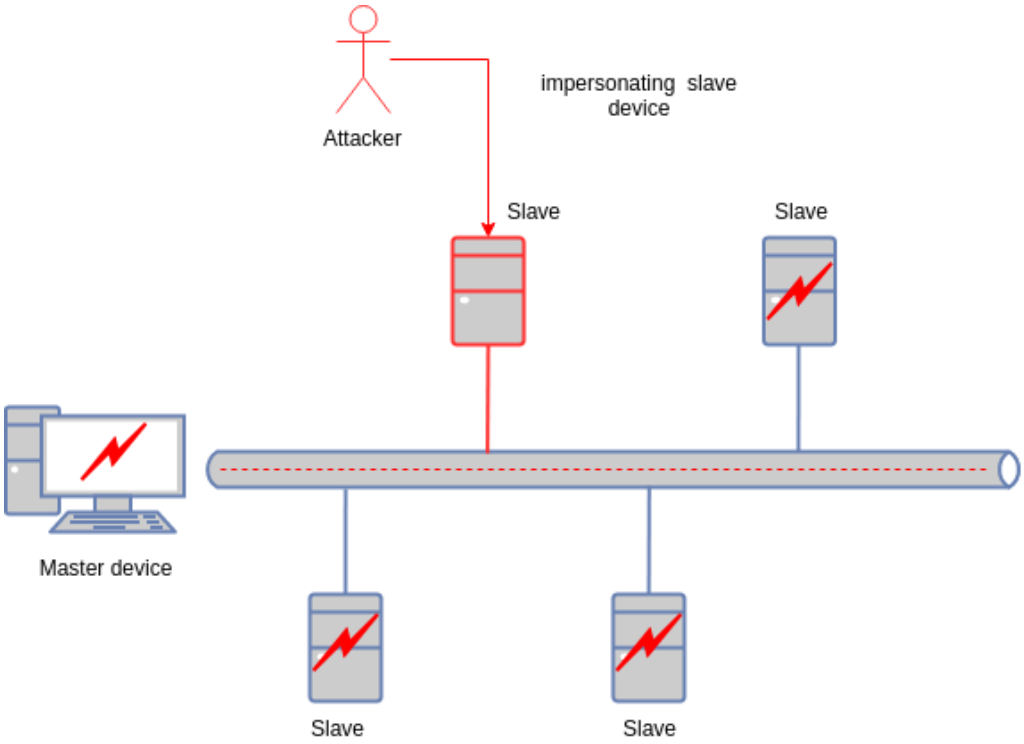

Figure 9: Schematic of the Man-in-the-Middle attack

5.1.4 PROFIsafe

It has been clear that the Ethernet standard is present in the industry for many years. Due to its popularity, there have been several protocols which target specifically designed requirements often referred to as Real-time Ethernet protocols [89]. PROFINET is one of the mentioned distribution of Ethernet protocol based on the IEEE 802 standard, developed by PROFIBUS International User Group. It can be described as a wire protocol with properties from which objects can be manipulated and connected [90]. PROFINET satisfied all challenges requested by automation technologies. PROFINET is standardized by IEC 61158 and IEC 61784 standards. PROFIBUS is open Fieldbus standard used for automatization by PROFIBUS International, while the PROFINET is based on industrial Ethernet. What can be considered as a characteristic of this protocol is that allows using UDP/IP protocol as a protocol of higher level for demanded data transfers. PROFISafe can be considered as an extension of PROFINET and PROFIBUS systems. This protocol used black channel concept, which means that communication is pursed via safety application, which is built on top of communication protocol.

In Figure 12 PROFIsafe security essentials requirements can be seen. PROFIsafe has socalled security zones. Therefore, only options to violate security requirements of PROFIsafe protocol is to get acccess to mentioned zones. Only options of of

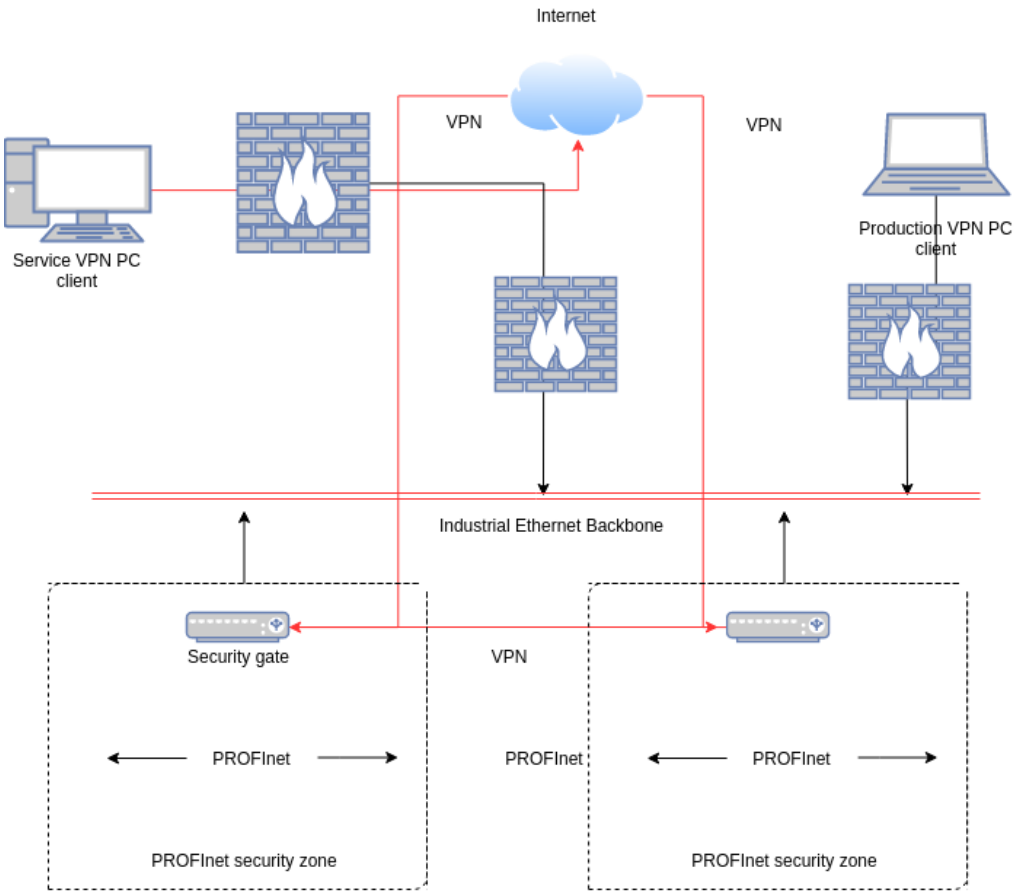

Figure 10: PROFIsafe and Security [91]

transfering the date is accepted throught the Virtual Private Network (VPN) and Firewalls [91]. According to the Akerberg et al. "it is possible to attack PROFIsafe and change the safety-related process data without the protocol detecting the attack" [92]. It is rather suprising that the authors came to this conclusion that PROFIsafe protocol standard can be manipulated without the detection, they preformed several attacks showing how to bypass all security measures. This scenario is possible if the malicious attacker has access to the physical network. According to the autors in real-world examples this attack scearios has low chance of occurence. Besides autors presented security framework to overcome this security issues. According to the [91] man in a middle attack attack is the most usefull in this scenarios beacuse of the likehood that the attack will be detacted is relatively small.

6 Description of chosen safety functions

In this section, we describe our approach that first enables modeling of safety functions using Rebeca model checker, and then enrich such an analysis by introducing security dimension in it. In this work we will use the example of an industrial robot. We choose to model and analyze two safety functions that are Control Error Supervision and Stand Still Supervision. In the following subsection, we extensively describe these functions and the possible scenario of their execution. We choose these two function as we are interested in their safety properties, but also envision them as good candidates to show the implications of potential security breaches to safety.

6.1 Control Error Supervision function

The control error supervision is a central part of the safety concept since it ensures that the reference and the measured position corresponds giving a dual channel input to the system. Whenever the robot is moved, the reference position and the actual position is compared. If the difference exceeds the tolerance the robot is stopped.

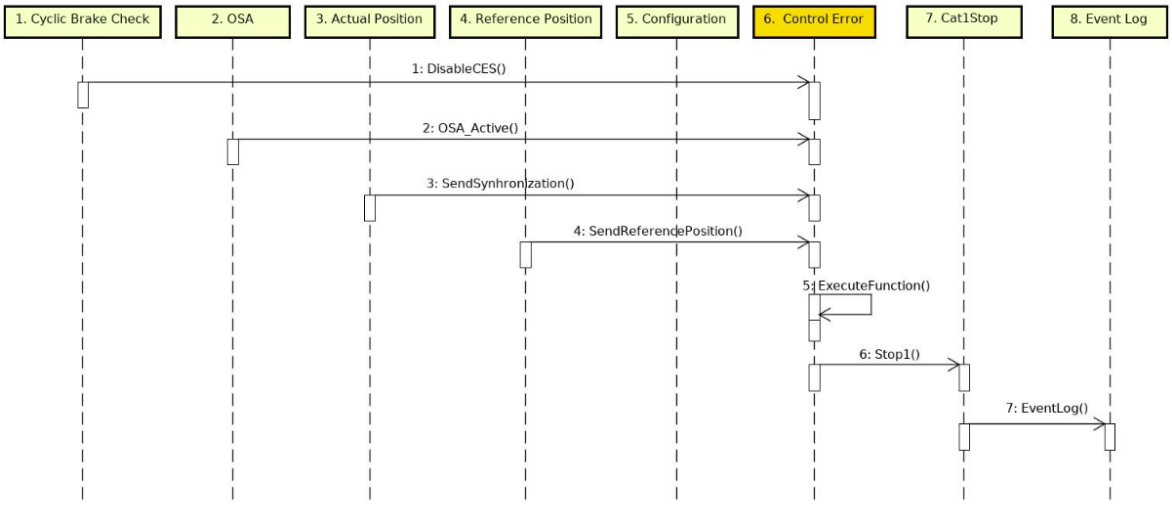

Figure 11: Control Error Supervision sequence diagram

6.1.1 Function properties

1. Cyclic Brake Check

   The cyclic brake check is a function that warns about and controlling brake checks. A brake check period is configured by the user, defining when brake checks are supposed to happen.

2. OSA (Operational Safety Area)

   The Operational Safety Area is another function which provides an input values to the Control Error Supervision. The OSA function is used when special

control modes of the robot are used (e.g. in drilling operation) where the servo loop gain is reduced (soft servo) allowing a greater control error to occur.

3. Actual Position

   The actual position is the current position of the robot, it is defined as angles of the motors at each joint.

4. Reference Position

   The reference position is the next position the robot shall be moved to. It is also defined as angles of the motors at each joint.

5. Configuration

   The Control Error Supervision function needs the following configuration information for each axis: Servo lag (radians on the motor side), Servo Delay, The Control Error Supervision function requires the control error tolerances for OSA, in order to decide which axes have relaxed supervision when OSA is active.

6. Control Error

   Control Error function receives values according to the specification and then accordingly triggers cat1 stop and sends an appropriate event log. This function ensures that the Reference and Actual position are synchronized.

7. Cat1Stop

   When Cat1 is activated the stop is initiated by the software on the Safety Controller that sends a stop request to the robot control application. Unlike the Cat0Stop which is characterized by removing the power, within this function stop request is sent to the Robot OS.

8. Event Log

   Event log accumulates and stores the data about the Control Error function. This output data is used in order to analyze the events that had occurred in the previous iteration.

6.1.2 Execution logic

One scenario of executing the Control Error Supervision function is presented in Figure 7. Cyclic brake check function first informs Control Error that DisableCES variable is set to false, therefore at the moment Cyclic brake is not active. Operational Safety area sends a signal that its property of OSA_Active = true which means that Control error is relaxed to a higher value. Furthermore, the actual position sends its current position and information about synchronization with Reference position. Reference position which is the next position the robot shall be moved to is sending information to the Control Error as well. Configuration which is usually defined by the user is sent to the Control Error. If Control error notices false values cat1 stop is triggered resulting that robot is in stop state and an appropriate event log is issued as well.

6.2 Stand Still Supervision function

During collaborative work between the user and the robot, it is important that the robot is in standing still mode. With this function the robot is working with servo and drive system still regulating, but having zero speed (robot/axes standing still). If any movement is detected when this function is active, a Cat0Stop stop shall be triggered.

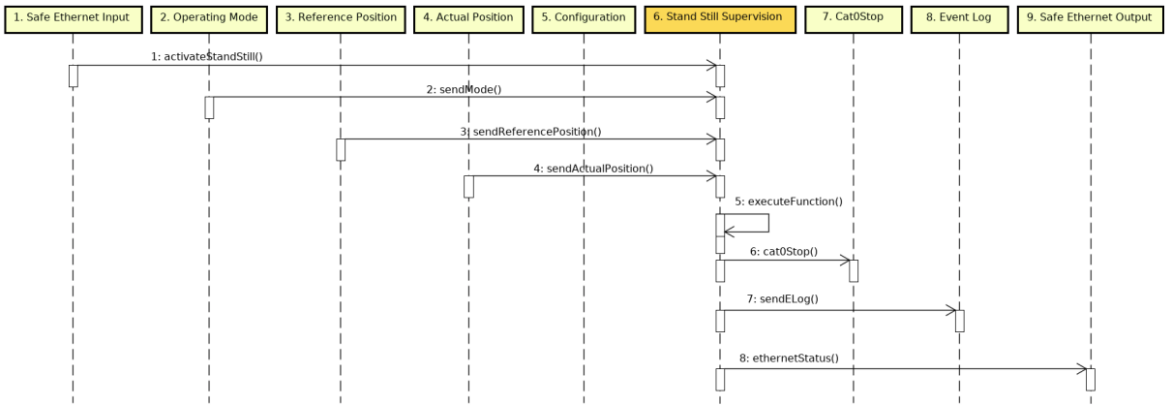

Figure 12: Stand Still Supervision sequence diagram

6.2.1 Function properties

1. Safe Ethernet Input

   This input data represents the signal used to activate the appropriate function. Basically, it is a Boolean value that triggers functionality in the safety system.

2. Operating Mode

   The system operates within three modes:

   - Manual FS
   - Manual
   - Auto

   There are two options for setting the operating mode, either a mechanical switch or a software implemented switch.

3. Reference position

   The reference position is the next position the robot shall be moved to. It is also defined as angles of the motors at each joint.

4. Actual Position

   The actual position is the current position of the robot, it is defined as angles of the motors at each joint.

5. Configuration

   The configuration of the function consists of a set of parameters where the AxisTolerance is the main parameter. It describes how much the robot is allowed to be moved when the function is active. The value corresponds to the angle of the axis. The activation function defines when the function should be active. Violation describes what should happen when the function is violated. Status function is in this case a possible signal indicating that the function is active.

6. Stand Still Supervision

   Based on the received values, Stand Stil Supervision function decides whether Cat0Stop shall be triggered, with appropriate event log and Safe Ethernet Output based on the prior values.

7. Cat0Stop

   This function allows stopping of the manipulator by removing power the power to the motors and applying the brakes. No configuration is needed for the function Cat0Stop.

8. Event Log

   The event log is just a a document with collected values of important robot parameters. For stand still supervision the message has the following parameters: Message number, Title, Description, Consequences, Causes, and Actions.

9. Safe Ethernet Output

   Similar to Safe Ethernet Input, this data describes a set of Boolean signals. In this case, signals carry information about the current active state of function and indicates whether the violation has occurred.

6.2.2 Execution logic

In the sequence diagram in Figure 8, information flow for Stand Still Supervision function can be observed. This function is triggered by receiving Safe Ethernet Input with the boolean value used for activation. Furthermore, Operating mode is chosen from three allowed modes (i.e. Manual FS, Manual, and Auto). Reference and Actual position signals are used to provide the values to Stand Still Supervison function in the same way as for the Control Error function. Configuration signal is activated by sending the expected values. Finally, if the Stand Still functions with all mentioned received values notice invalid values or false synchronization, Cat0Stop will be triggered. Therefore, this function will automatically remove power from the system, and Safe Ethernet Output will send values accordingly.

7 Adding security dimension to safety functions

In this section, we describe our reasoning about security and the way we include it in already existing safety functions. Furthermore, we incorporate 4 attack scenarios in order to analyze in case if a malicious user gains unauthorized access. It is important to emphasize that many companies nowadays migrate their software solution to cloud technologies, that might make their software more vulnerable. In these examples, we propose 4 attacks that could potentially seriously harm system properties or even have safety implications. Our idea is to show how the attacker potentially can use legitimate ways of communication in order to jeopardize system properties. We verify these scenarios using Rebeca language.

It is necessary to emphasize that we assume take a scenario where a security breach in the system already occurred. Security breach by remote access or simply by plugging the cable into a machine within the manufacturing facilities and taking control over the system is also possible. In these scenarios, we point out how in some cases system owners might not be aware of the security breach until the system is completely compromised. Due to the nature of robotics and manufacturing, many people might possibly gain access to the system and it is logical to assume that chances for security breaches will increase.

7.1 An attack scenarios for CES

The shapes means the following in the diagrams:

- Rectangle: functions
- Parallelogram: in/out data

7.1.1 Attack Scenario 1: OSA manipulation

In order to compromise the system, attacker might first communicate with OSA function by changing its boolean active value from *false* to *true*. Thus, if OSA is always active that implies that safety in some sense is always reduced. Therefore, when the system is already compromised previously defined Configuration values can be manipulated as well. For instance, max servo lag value can be set to a false value which will eventually manipulate robot position.

As a reminder, if the safety controller is in the unsynchronized state, no supervision of the control error is done. However, in this case, reference and actual position are synchronized. Therefore, the Control Error function is unable to detect this attack. In Figure 14 graphical representation of attack can be observed. In Figure 15 sequence diagram of this attack is presented as well.

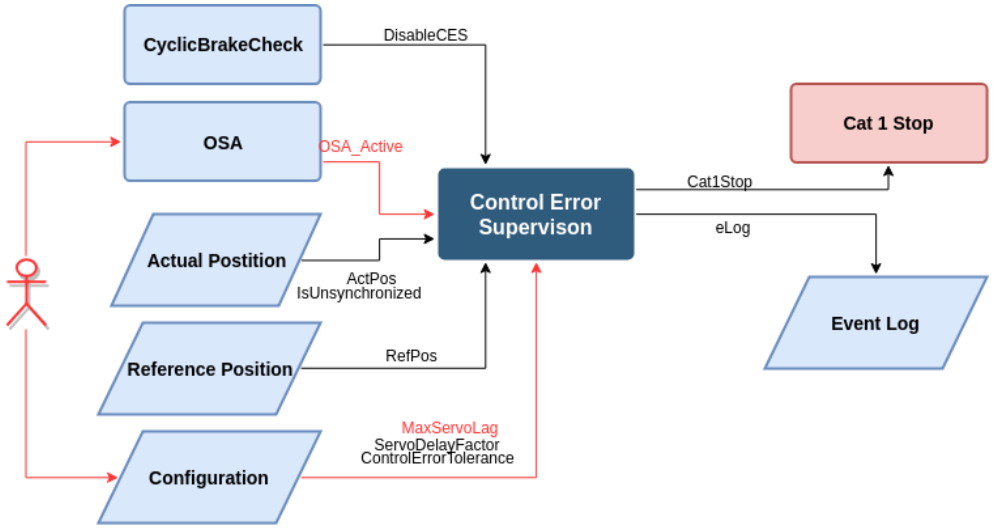

Figure 13: Attack Scenario 1: OSA manipulation

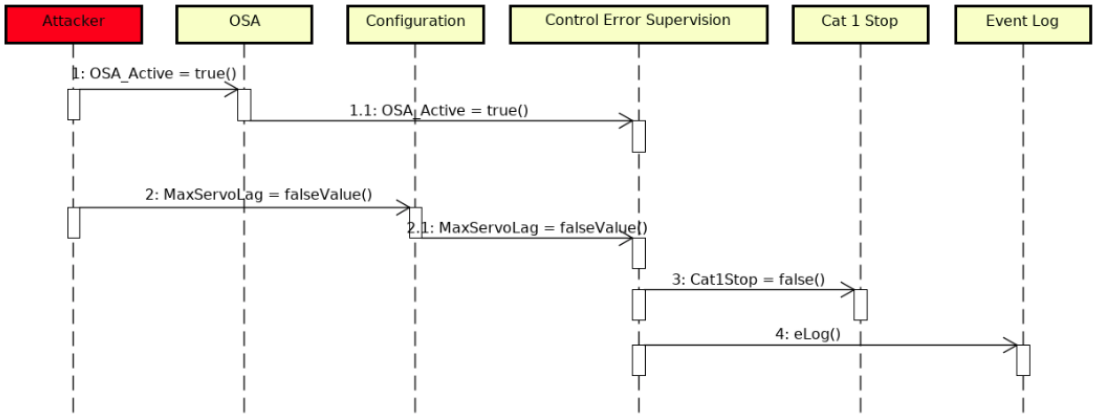

Figure 14: OSA manipulation sequence diagram

7.1.2 Attack Scenario 2: Cyclic Brake Check manipulation

In this attack Cyclic Brake Check value is compromised. It is useful to show how by incorporating this attack system can be compromised. Furthermore, based on requirements when Cyclic Brake Check is active the Control Error Supervision is disabled during the test in order to reduce false alarms. Therefore, we cannot see any other redundant or basic mechanism to prevent Control Error Supervision to be triggered when Cyclic Brake is active. Thus, in this attack we are able to manipulate current boolean value from *false* to *true*. Therefore, in this scenario, the safety aspect is not jeopardized. However, RobotOS will trigger Cat1Stop leading to decreased productivity.

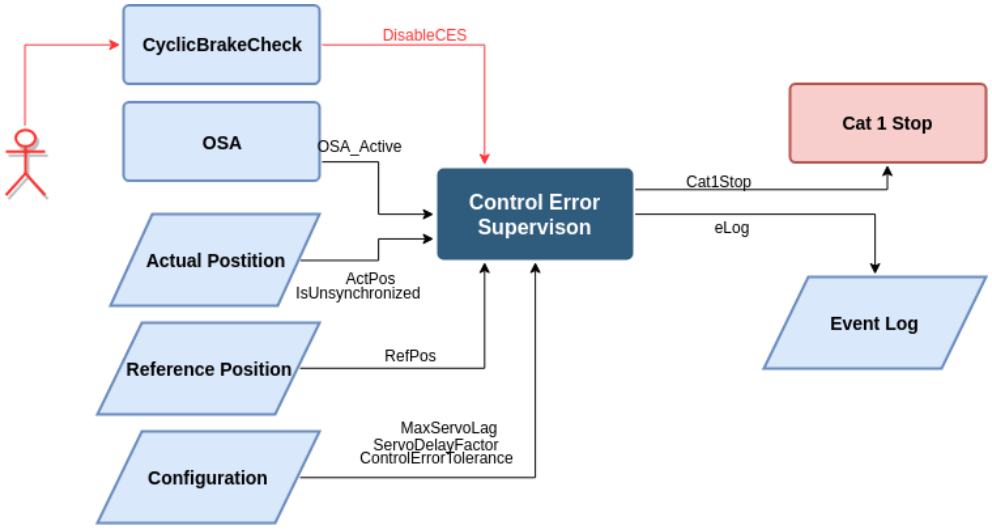

Figure 15: Attack Scenario 2: Cyclic Brake Check manipulation

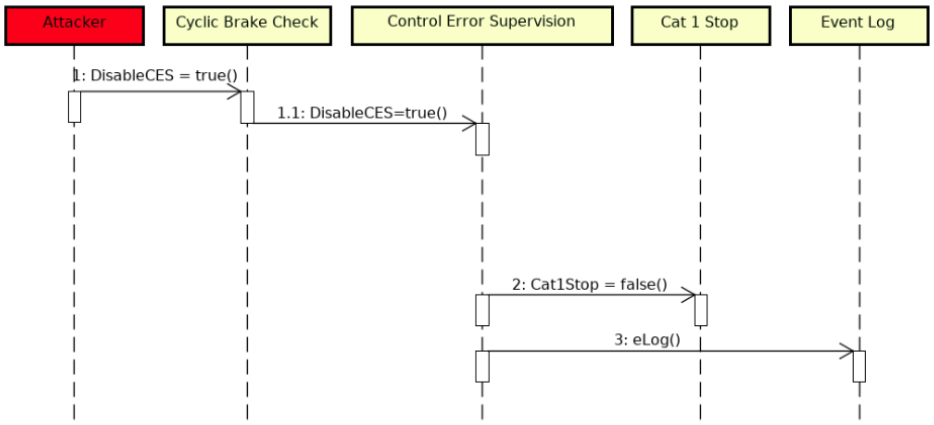

Figure 16: Attack Scenario 2 sequence diagram

7.2 An attack scenarios for SSS

7.2.1 Attack Scenario 1: Operating mode manipulation

In this scenario, we construct an attack on the Stand Still Supervision function. Furthermore, Operating Mode values will be compromised. According to the specification, there are three specified and allowed modes i.e. FS Manual, Manual, and Automatic. Our idea is to simply provide unspecified value of operating mode to the system, or just manipulate its current status. This scenario could have potentially serious consequences for nearby workers, as by conducting this attack robot will no longer be in Stand Still position.

7.2.2 Attack Scenario 2: Position manipulation

In this case, our idea is to compromise the Actual position by providing false values. Actual position sends two values to the function, i.e. current position and

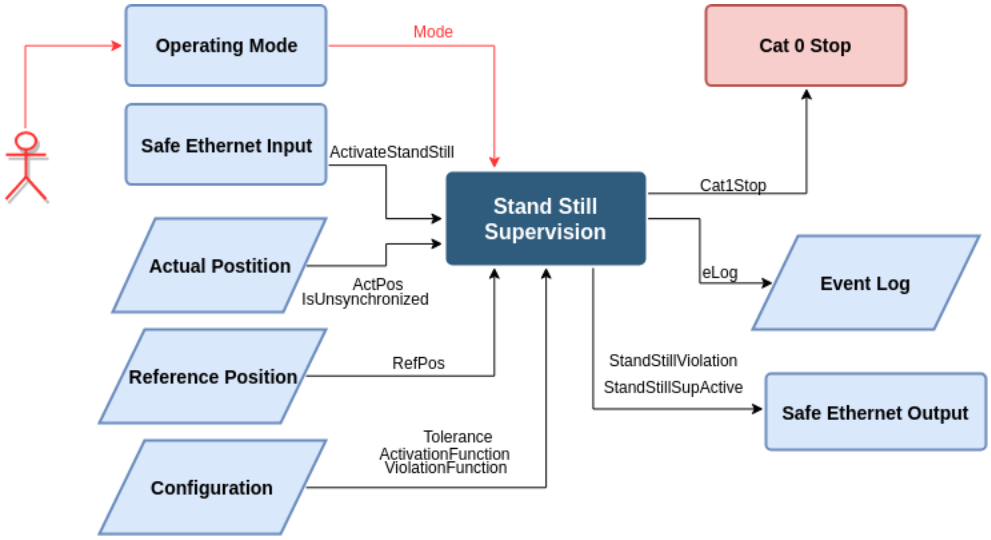

Figure 17: Attack Scenario 1 - Operating mode manipulation

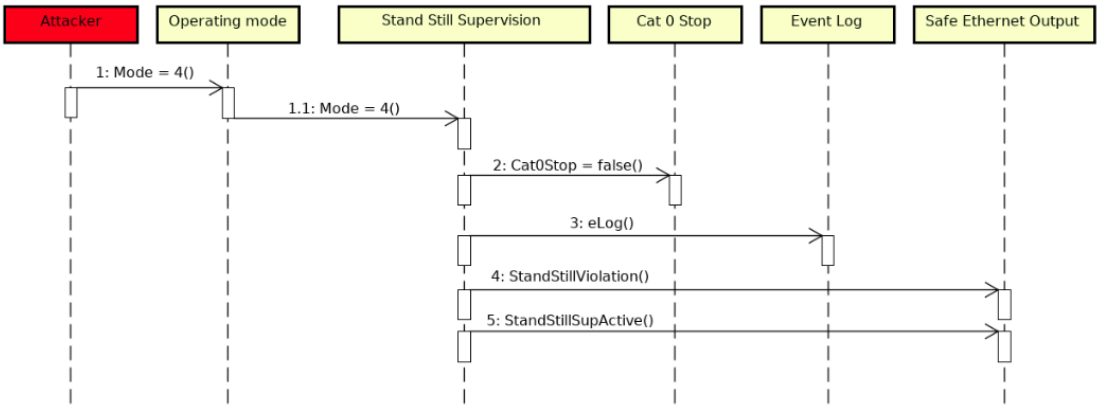

Figure 18: Operating mode manipulation sequence diagram

IsUnsynchronized which is the boolean value used to determine is it Actual position corresponding to the Reference position. Therefore, by providing false value for the ActPos variable, isUnsynchronized will be triggered. However, in order to make manipulation more complex, we are changing its value from *false* to *true*. Whit that manipulation Stand Still Supervision function will not detect this attack. Moreover, Safe Ethernet Output will be making this function active through StandStillActive variable.

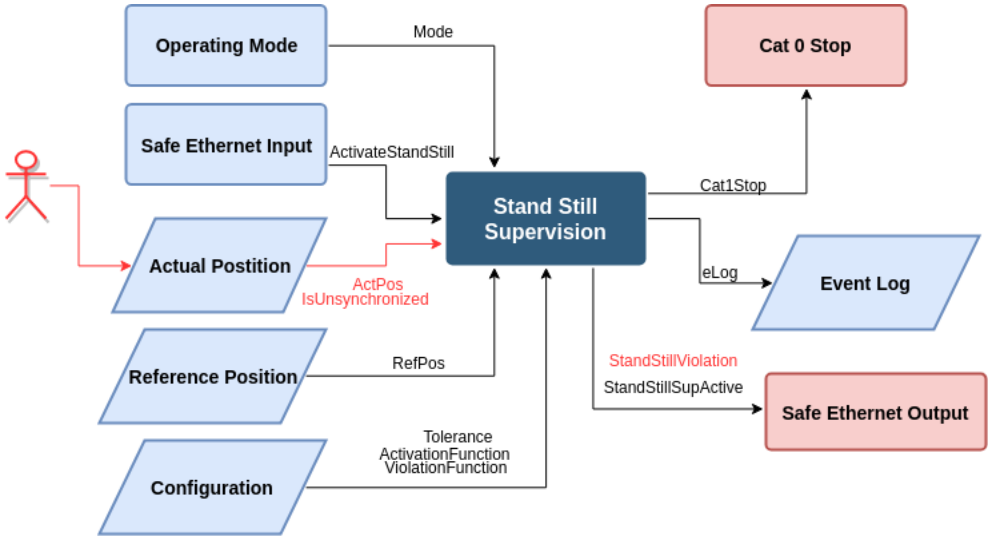

Figure 19: Attack Scenario 2: Position manipulation

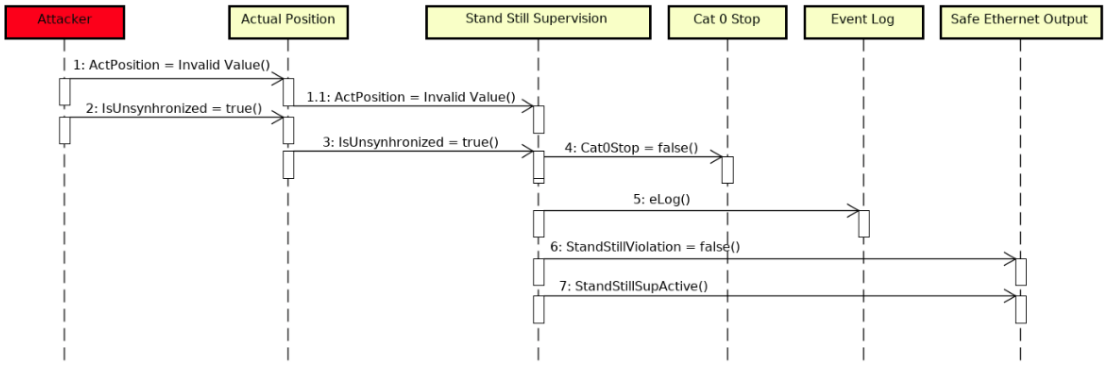

Figure 20: Position manipulation attack sequence diagram

7.3 Possible strategies for mitigation and prevention

In order to potentially prevent possible attacks or enhance system security in this subsection possible strategies for mitigation or total prevention of attacks will be discussed. From our point of view in order to prevent potentially harmful consequences to the system, we propose a redundant mechanism for the sensitive functions that are responsible for robot movement.

7.3.1 Control Error Supervision proposed changes

Graphical representation of our proposed changes for the CES can be observed in Figure 13. Our idea is to design a redundant mechanism in order to gain more control over the functions required for robot movement. Moreover, Actual and Reference position sends data to the Redundant mechanism and Control Error Supervision simultaneously. After Redundant mechanism received the data, isUnsynhornized value is issue to the CES. Therefore, after CES received value, the

isUnsynhrnoized variable is compared to the directly received value from the Reference and Actual position. Values provided to the Redundant mechanism should have less timing frequency compared to the CES. This mechanism will also control current value from OSA and CyclicBrakeCheck as we believe that these values are very important for the execution of the function. Therefore, by incorporating this mechanism our attacks scenarios to the CES function can be mitigated or fully prevent.

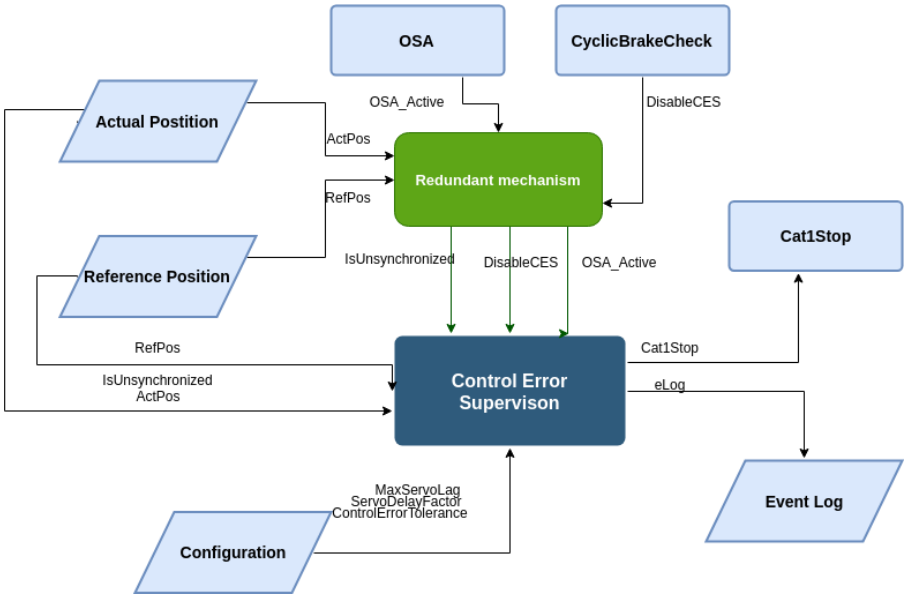

Figure 21: Control Error changes

7.3.2 Stand Still Supervision proposed changes

For the case of SSS function, the same logic is applied as for the CES regarding position manipulation concerning attack scenario 2. For the attack scenario 1, we tend to believe that our proposed attack can be prevented by proper code implementation. Furthermore, if the software is implemented in such a way that insertion of unspecified mode (there are 3 specified modes according to the specification i.e. FS Manual, Manual, and Automatic) will only notify the user with an appropriate event log, then our attack will be prevented.

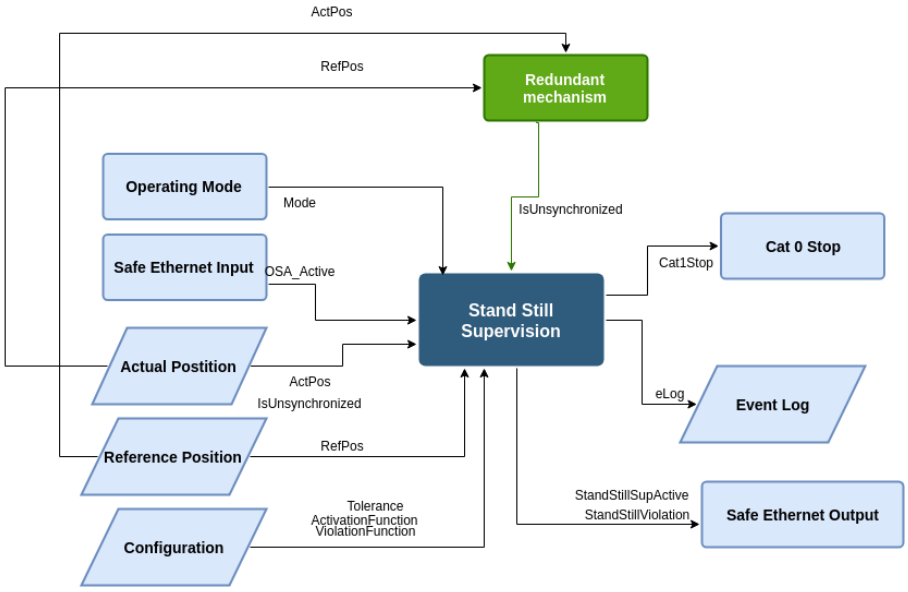

Figure 22: Stand Still changes

8 Discussion

In the following section the research questions proposed in Section 3 are revisited.

RQ-1:What existing approaches are addressing security by means of model-checking techniques?

In order to provide an answer to this research question, we conducted extensive literature survey resulting in initial set of 18892 papers regarding model checking approaches used to analyze security.

After reading titles and abstracts of those papers we have been able to reduce the initial number to 43 papers, as discussed in Section 4. We have been able to see that there is a significant number of publication presenting approaches that are based on formal methods and model checking that are addressing different security issues. Some of them are more popular, like Spin, while some of them have rather limited number of publications in this domain, like Rebeca.

RQ-2: How to model security by using model checking approach in safety critical applications?

During our research, we focus on the context of safety-critical application such as software of industrial robot. Therefore, we first modeled two highly sensitive functions from the safety aspect by using Rebeca model checker. Moreover, we further expand the modeling aspect by adding two security scenario for each function. We model security in such a way that Attacker can be observed as one more reactive class in the whole system and therefore it can use legitimate channels for communication. In order to use required timing requirements within the function execution, we used Timed Rebeca which is Rebeca distribution that supports the modeling and verification of distributed systems with timing features.

RQ-2a: What are the possible implications on system safety that can

be identified by modeling security?

We have identified some potential system vulnerabilities which could cause problems on safety. Moreover, modeling security aspects in the safety-critical applications should have highest priority level during development process. From our attacks scenarios it is clear that by jeopardizing safety aspect is difficult task even when system is compromised. However in some special cases such as during execution of Stand Still Supervision function safety aspects can be manipulated by security breaches as we show in our attack scenarios. In order to enhance the security aspects of the system, we proposed possible strategies for mitigation or complete prevention of these kinds of attacks.

9 Conclusions

It is evident that in order to implement an entirely reliable system, security and safety features needs to be tightly interconnected. Hence, due to the risk of security breaches safety-critical systems are expected to respond to potentially challenging security aspects. There is no completely security prone system, as security is dynamic by its nature and new threats and vulnerabilities might evolve during the system life time. We have to be aware that if security is not taken into consideration in systems, especially those that are safety-critical, a potentially sever consequences might occur.

To fully understand these issues and present findings we conducted extensive literature survey about existing work in the topic of security and model-checking approaches. As a result, we got 43 highly relevant papers and their essential information are briefly described. Taking mentioned literature survey in the account we can conclude that this approach of dealing with security issues has multiple benefits such as uncovering all possible traces of the function execution where security could provide an impact of function's expected behavior. With the trend of an increased number of publications, we can also point out that modeling security with model checking tools is becoming more popular.

Furthermore, we provide a model and analysis of an industrial robot functions using Rebeca model checker. We chose two functions that are essential from the safety-critical point of view and therefore they serves as a great example to which part of the system security attacks should be incorporated. We provided a sequence diagram for each function and its properties and logic are described as well.

In order to demonstrate how security can jeopardize safety, we propose 4 attacks scenarios to the provided case study. Therefore, in order to mitigate or fully prevent this kind of attacks we presented and elaborated our approach in order to improve the security aspects of the system.

It is important to emphasize that, even do the attacks are identified as relevant and possible, there is a small possibility that the attacker will endanger the safety aspect by security breaches.

Model checking approach shown to be useful in modeling potential security implication, as we have been able to uncover potential problems in safety when security has been taken into account. Using such an approach might lower the cost of implementation, but on the other hand it might be tedious for engineers to use and apply such an approach without appropriate knowledge and training.

References

[1] D. F. Brewer, "Applying security techniques to achieving safety," in *Directions in Safety-Critical Systems*. Springer, 1993, pp. 246–256.

[2] D. P. Eames and J. Moffett, "The integration of safety and security requirements," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 1999, pp. 468–480.

[3] K. Hänninen, H. Hansson, H. Thane, and M. Saadatmand, "Inadequate risk analysis might jeopardize the functional safety of modern systems," *arXiv preprint arXiv:1808.10308*, 2018.

[4] Hackers can remotely access syringe infusion pumps to deliver fatal overdoses. https://thehackernews.com/2017/09/hacking-infusion-pumps.html. Accessed: 2018-12-10.

[5] Statement from fda commissioner scott gottlieb, m.d. on fda's efforts to strengthen the agency's medical device cybersecurity program as part of its mission to protect patients. https://www.fda.gov/NewsEvents/Newsroom/PressAnnouncements/ucm622074.htm. Accessed: 2018-12-10.

[6] C. Schmittner, Z. Ma, and P. Smith, "Fmvea for safety and security analysis of intelligent and cooperative vehicles," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2014, pp. 282–288.

[7] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner, "Sahara: a security-aware hazard and risk analysis method," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 2015, pp. 621–624.

[8] E. M. Clarke, E. A. Emerson, and J. Sifakis, "Model checking: algorithmic verification and debugging," *Communications of the ACM*, vol. 52, no. 11, pp. 74–84, 2009.

[9] H. R. Shahriari, M. S. Makarem, M. Sirjani, R. Jalili, and A. Movaghar, "Vulnerability analysis of networks to detect multiphase attacks using the actor-based language rebeca," *Computers & Electrical Engineering*, vol. 36, no. 5, pp. 874–885, 2010.

[10] J. C. Knight, "Safety critical systems: challenges and directions," in *Proceedings of the 24th International Conference on Software Engineering*. ACM, 2002, pp. 547–550.

[11] A. Albee, S. Battel, R. Brace, G. Burdick, J. Casani, J. Lavell, C. Leising, D. MacPherson, P. Burr, and D. Dipprey, "Report on the loss of the mars polar lander and deep space 2 missions," 2000.

[12] M. I. Board, "Mars climate orbiter mishap investigation board phase i report november 10, 1999," 1999.

[13] P. G. Neumann, *Computer-related risks*. Addison-Wesley Professional, 1994.

[14] I. Sljivo, "Assurance aware contract-based design for safety-critical systems," Ph.D. dissertation, Mälardalen University, 2018.

[15] R. Kissel, *Glossary of key information security terms*. Diane Publishing, 2011.

[16] D. Hanic and A. Surkovic, "An attack model of autonomous systems of systems," 2018.

[17] A. Greenberg, "The jeep hackers are back to prove car hacking can get much worse," *WIRED article, https://www. wired. com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks*, 2016.

[18] Attack modeling vs threat modeling. https://www.techrepublic.com/article/attack-modeling-vs-threat-modeling/. Accessed: 2018-12-10.

[19] P. L. Goddard, "Software fmea techniques," in *Reliability and Maintainability Symposium, 2000. Proceedings. Annual*. IEEE, 2000, pp. 118–123.

[20] C. Hewitt, "Description and theoretical analysis (using schemata) of planner: A language for proving theorems and manipulating models in a robot," MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, Tech. Rep., 1972.

[21] G. A. Agha, "Actors: A model of concurrent computation in distributed systems." MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, Tech. Rep., 1985.

[22] G. Agha, *An overview of actor languages*. ACM, 1986, vol. 21, no. 10.

[23] K. Hansen, "Security attack analysis of safety systems," in *Emerging Technologies & Factory Automation, 2009. ETFA 2009. IEEE Conference on*. IEEE, 2009, pp. 1–4.

[24] Story of jonathan james who hacked nasa and pentagon in age of 15. https://techasli.com/story-jonathan-james-hacked-nasa-pentagon-age-15/. Accessed: 2018-12-10.

[25] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, no. 2004, pp. 1–26, 2004.

[26] K. G. Larsen, P. Pettersson, and W. Yi, "Uppaal in a nutshell," *International journal on software tools for technology transfer*, vol. 1, no. 1-2, pp. 134–152, 1997.

[27] R. Corin, S. Etalle, P. H. Hartel, and A. Mader, "Timed model checking of security protocols," in *Proceedings of the 2004 ACM workshop on Formal methods in security engineering*. ACM, 2004, pp. 23–32.

[28] G. Díaz, F. Cuartero, V. Valero, and F. Pelayo, "Automatic verification of the tls handshake protocol," in *Proceedings of the 2004 ACM symposium on Applied computing*. ACM, 2004, pp. 789–794.

[29] S. Mondal, S. Sural, and V. Atluri, "Towards formal security analysis of GTRBAC using timed automata," in *Proceedings of the 14th ACM symposium on Access control models and technologies - SACMAT '09*, ser. SACMAT '09. ACM, 2009, p. 33.

[30] K. Saghar, D. Kendall, and A. Bouridane, "Raeed: A solution for hello flood attack," in *Applied Sciences and Technology (IBCAST), 2015 12th International Bhurban Conference on*. IEEE, 2015, pp. 248–253.

[31] I. El Kassmi, Z. Jarir, I. E. Kassmi, Z. Jarir, I. El Kassmi, and Z. Jarir, "Security requirements in web service composition: Formalization, integration, and verification," in *Proceedings - 25th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2016*. IEEE, jun 2016, pp. 179–184.

[32] S. Amin, K. Saghar, M. B. T. Abbasi, and A. Elahi, "ARHUM-ARRIVE protocol with Handshake Utilization and Management," in *Proceedings of 2017 14th International Bhurban Conference on Applied Sciences and Technology, IBCAST 2017*, 2017, pp. 401–407.

[33] Y. Sun, T.-Y. Wu, X. Ma, and H.-C. Chao, "Modeling and verifying epc network intrusion system based on timed automata," *Pervasive and Mobile Computing*, vol. 24, pp. 61–76, 2015.

[34] O. Gadyatskaya, R. R. Hansen, K. G. Larsen, A. Legay, M. C. Olesen, and D. B. Poulsen, "Modelling attack-defense trees using timed automata," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9884 LNCS, pp. 35–50.

[35] G. J. Holzmann, "The model checker spin," *IEEE Transactions on software engineering*, vol. 23, no. 5, pp. 279–295, 1997.

[36] A. Jøsang, "Security protocol verification using spin," 1995.

[37] R. X. CCITT, "509,"the directory—authentication framework," consultation committee, international telephone and telegraph," *International Telecommunications Union, Geneva*, 1989.

[38] P. Maggi and R. Sisto, "Using spin to verify security properties of cryptographic protocols," in *International SPIN Workshop on Model Checking of Software*. Springer, 2002, pp. 187–204.

[39] P. K. Singh and A. Lakhotia, "Static verification of worm and virus behavior in binary executables using model checking," in *Information Assurance Workshop, 2003. IEEE Systems, Man and Cybernetics Society*. IEEE, 2003, pp. 298–300.

[40] S. Nakajima, "Model-checking of safety and security aspects in web service flows," in *International Conference on Web Engineering*. Springer, 2004, pp. 488–501.

[41] M. Xiao and J. Li, "The Modeling Analysis of Cryptographic Protocols using Promela," in *2006 6th World Congress on Intelligent Control and Automation*, vol. 1. IEEE, 2006, pp. 4321–4325.

[42] Y. Jiang and X. Liu, "Formal Analysis for Network Security Properties on a Trace Semantics," in *2008 International Conference on Advanced Computer Theory and Engineering*, 2008, pp. 957–960.

[43] W. Wang and D. Ji, "Using SPIN to detect vulnerabilities in the AACS drive-host authentication protocol," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008, vol. 5048 LNCS, pp. 305–323.

[44] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on information theory*, vol. 29, no. 2, pp. 198–208, 1983.

[45] J. J. Li and J. J. Li, "Model Checking the SET Purchasing Process Protocol with SPIN," in *2009 5th International Conference on Wireless Communications, Networking and Mobile Computing*, no. 70871063. IEEE, 2009, pp. 1–4.

[46] J. Ma, D. Zhang, G. Xu, and Y. Yang, "Model Checking Based Security Policy Verification and Validation," in *2010 2nd International Workshop on Intelligent Systems and Applications*, 2010, pp. 1–4.

[47] S. Basagiannis, P. Katsaros, and A. Pombortsis, "An intruder model with message inspection for model checking security protocols," *Computers and Security*, vol. 29, no. 1, pp. 16–34, feb 2010.

[48] W. Chen and W. Xiao, "Model checking and analyzing the security protocol for wireless sensor networks," in *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, vol. 8, 2011, pp. 4093–4096.

[49] S. Basagiannis, P. Katsaros, and A. Pombortsis, "Synthesis of attack actions using model checking for the verification of security protocols," *Security and Communication Networks*, vol. 4, no. 2, pp. 147–161, feb 2011.

[50] X. Chang, B. Xing, and Y. Qin, "Formal analysis of a response mechanism for TCG TOCTOU attacks," in *Proceedings - 2012 4th International Conference on Multimedia and Security, MINES 2012*, no. 2. IEEE, nov 2012, pp. 19–22.

[51] N. G. Kushik, A. Mammar, A. Cavalli, N. V. Yevtushenko, W. Jimenez, and E. Montes de Oca, "A SPIN-based approach for detecting vulnerabilities in C programs," *Automatic Control and Computer Sciences*, vol. 46, no. 7, pp. 379–386, dec 2012.

[52] S. Chen, H. Fu, and H. Miao, "Formal verification of security protocols using Spin," in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*. IEEE, 2016.

[53] S. Madhusudhanan and S. Mallissery, "Provable security analysis of complex or smart computer systems in the smart grid," in *2017 IEEE International Conference on Smart Grid and Smart Cities (ICSGSC)*, 2017, pp. 210–214.

[54] M. Kwiatkowska, G. Norman, and D. Parker, "Prism: Probabilistic symbolic model checker," in *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*. Springer, 2002, pp. 200–204.

[55] Probabilistic model checking. https://www.prismmodelchecker.org/lectures/pmc/06-prism.pdf. Accessed: 2019-02-10.

[56] G. Steel, "Formal analysis of PIN block attacks," *Theoretical Computer Science*, vol. 367, no. 1-2, pp. 257–270, nov 2006.

[57] S. Basagiannis, P. Katsaros, A. Pombortsis, and N. Alexiou, "A Probabilistic Attacker Model for Quantitative Verification of DoS Security Threats," in *2008 32nd Annual IEEE International Computer Software and Applications Conference*, 2008, pp. 12–19.

[58] T. Deshpande, P. Katsaros, S. Basagiannis, and S. A. Smolka, "Formal Analysis of the DNS Bandwidth Amplification Attack and Its Countermeasures Using Probabilistic Model Checking," in *2011 IEEE 13th International Symposium on High-Assurance Systems Engineering*, 2011, pp. 360–367.

[59] G. Zonghao, W. Ou, Y. Peng, H. Lansheng, and W. Weiming, in *2016 IEEE Trustcom/BigDataSE/ISPA*.

[60] B. Huang, Y. Huang, J. Kong, and X. Huang, "Model Checking Quantum Key Distribution Protocols," in *2016 8th International Conference on Information Technology in Medicine and Education (ITME)*, 2016, pp. 611–615.

[61] N. Alexiou, S. Basagiannis, and S. Petridou, "Formal security analysis of near field communication using model checking," *Computers and Security*, vol. 60, pp. 1–14, jul 2016.

[62] T. Chen, T. Han, F. Kammueller, I. Nemli, and C. W. Probst, "Model based analysis of insider threats," in *2016 International Conference On Cyber Security And Protection Of Digital Services (Cyber Security)*, 2016, pp. 1–3.

[63] M. Sirjani, A. Movaghar, A. Shali, and F. S. De Boer, "Modeling and verification of reactive systems using rebeca," *Fundamenta Informaticae*, vol. 63, no. 4, pp. 385–410, 2004.

[64] M. Sirjani and M. M. Jaghoori, "Ten years of analyzing actors: Rebeca experience," in *Formal Modeling: Actors, Open Systems, Biological Systems*. Springer, 2011, pp. 20–56.

[65] F. D. Boer, V. Serbanescu, R. Hähnle, L. Henrio, J. Rochas, C. C. Din, E. B. Johnsen, M. Sirjani, E. Khamespanah, K. Fernandez-Reyes *et al.*, "A survey of active object languages," *ACM Computing Surveys (CSUR)*, vol. 50, no. 5, p. 76, 2017.

[66] K. L. McMillan, "Symbolic model checking," in *Symbolic Model Checking*. Springer, 1993, pp. 25–60.

[67] L. Xia, H. Qi, and C. Yong, "Model Checking of Wireless Transaction Protocol," in *2009 WRI World Congress on Computer Science and Information Engineering*, vol. 7, 2009, pp. 620–623.

[68] R. W. Ritchey and P. Ammann, "Using model checking to analyze network vulnerabilities," in *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, 2000, pp. 156–165.

[69] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri, "Nusmv: A new symbolic model verifier," in *International conference on computer aided verification*. Springer, 1999, pp. 495–499.

[70] S. Adyanthaya, S. Rukmangada, A. Tiwari, and S. Singh, "Modeling freshness concept to overcome Replay attack in Kerberos protocol using NuSMV," in *2010 International Conference on Computer and Communication Technology (ICCCT)*, 2010, pp. 125–129.

[71] P. Mundra, M. Sharma, S. Shukla, and S. Singh, "Modeling and verification of Inter Realm Authentication in Kerberos using symbolic model verifier," in *Communications in Computer and Information Science*, 2011, vol. 204 CCIS, pp. 496–506.

[72] T. Jacob, M. Raman, and S. Singh, "Intrusion detection in zero knowledge system using model checking approach," in *Lecture Notes in Electrical Engineering*, 2013, vol. 131 LNEE, pp. 453–465.

[73] "Formalizing and Verification of an Antivirus Protection Service using Model Checking," *Procedia Computer Science*, vol. 57, pp. 1324–1331, 2015.

[74] S. Yamaguchi and M. S. B. A. Malek, "A Model Checking-Based Analysis Method of Cyber Attack in IoT System by Agent-Oriented Petri Net," in *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*, 2018, pp. 581–584.

[75] R. Shrestha, "Model Checking of Security Properties in Industrial Control Systems (ICS)," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '18. ACM, 2018, pp. 164–166.

[76] T. Gibson-Robinson, P. Armstrong, A. Boulgakov, and A. W. Roscoe, "Fdr3—a modern refinement checker for csp," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2014, pp. 187–201.

[77] G. Lowe, "Breaking and fixing the needham-schroeder public-key protocol using fdr," in *International Workshop on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 1996, pp. 147–166.

[78] S. Lu and S. A. Smolka, "Model checking the secure electronic transaction (SET) protocol," in *MASCOTS '99. Proceedings of the Seventh International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 1999, pp. 358–364.

[79] C. Kraetzer, R. Merkel, R. Altschaffel, E. Clausing, M. Schott, and J. Dittmann, "Modelling watermark communication protocols using the CASPER modelling language," in *Proceedings of the 12th ACM workshop on Multimedia and security - MM&Sec '10*. New York, NY, USA: ACM, 2010, p. 107.

[80] Z. Drias, A. Serhrouchni, and O. Vogel, "Taxonomy of attacks on industrial control protocols," in *2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS)*. IEEE, 2015, pp. 1–6.

[81] S. East, J. Butts, M. Papa, and S. Shenoi, "A taxonomy of attacks on the dnp3 protocol," in *International Conference on Critical Infrastructure Protection*. Springer, 2009, pp. 67–81.

[82] M. Majdalawieh, F. Parisi-Presicce, and D. Wijesekera, "Dnpsec: Distributed network protocol version 3 (dnp3) security framework," in *Advances in Computer, Information, and Systems Sciences, and Engineering*. Springer, 2007, pp. 227–234.

[83] CNN, "Mouse click could plunge city into darkness, experts say," *CNN, https://news.joins.com/article/2896348*, 2007.

[84] D.-g. Peng, H. Zhang, L. Yang, and H. Li, "Design and realization of modbus protocol based on embedded linux system," in *2008 International Conference on Embedded Software and Systems Symposia*. IEEE, 2008, pp. 275–280.

[85] P. Huitsing, R. Chandia, M. Papa, and S. Shenoi, "Attack taxonomies for the modbus protocols," *International Journal of Critical Infrastructure Protection*, vol. 1, pp. 37–44, 2008.

[86] I. N. Fovino, A. Carcano, M. Masera, and A. Trombetta, "Design and implementation of a secure modbus protocol," in *International conference on critical infrastructure protection*. Springer, 2009, pp. 83–96.

[87] A. Carcano, I. N. Fovino, M. Masera, and A. Trombetta, "Scada malware, a proof of concept," in *International Workshop on Critical Information Infrastructures Security*. Springer, 2008, pp. 211–222.

[88] O. Eigner, P. Kreimel, and P. Tavolato, "Detection of man-in-the-middle attacks on industrial control networks," in *2016 International Conference on Software Security and Assurance (ICSSA)*. IEEE, 2016, pp. 64–69.

[89] G. Prytz, "A performance analysis of ethercat and profinet irt," in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE, 2008, pp. 408–415.

[90] P. Ferrari, A. Flammini, D. Marioli, and A. Taroni, "Experimental evaluation of profinet performance," in *IEEE International Workshop on Factory Communication Systems, 2004. Proceedings.* IEEE, 2004, pp. 331–334.

[91] F. Reichenbach, J. Endresen, M. M. Chowdhury, and J. Rossebø, "A pragmatic approach on combined safety and security risk analysis," in *2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops*. IEEE, 2012, pp. 239–244.

[92] J. Åkerberg and M. Björkman, "Exploring network security in profisafe," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2009, pp. 67–80.