

Security Verification and Analysis

Effective Verification Approaches in Complex IoT Environments to Ensure their Security

Date: October 2020

Editor: Heribert Vallant

Authors: Heribert Vallant, Christian Lettner, Arndt Bonitz, Faiq Khalid,
Muhammad Shafique, Stefan Mangard, Martin Matschnig,
Katharina Pfeffer, Rudolf Ramler, Edin Arnautovic, Mario Lamberger



Contents

1. Summary	3
2. Challenge	4
2.1. Automated Driving	6
2.2. Smart Production/I4.0	6
3. Current Status	7
3.1. Protection Profile	7
3.2. Formal Verification of Side-Channel Protected Hardware Implementation	7
3.3. Hardware Property Checks	8
3.4. Dynamic Exchangeable Runtime Checkers in HW	8
3.5. Threat Modelling	8
3.6. Testing	9
3.7. Human Aspects	9
3.8. Analytical Toolbox	10
3.9. Formal Analysis of the Integrated Circuits	10
3.10. Anomaly Detection in Vehicular ad/hoc Network	10
3.11. IoT Discovery and Classification	11
4. Results	12
4.1. Automotive Ethernet Protection Profile	12
4.2. Formal Verification of Side/Channel Protected Hardware	12
4.3. Hardware Property Checks	13
4.4. Dynamic Exchangeable Runtime Checkers in HW	13
4.5. Threat Modelling	14
4.6. Testing	14
4.7. Human Aspects	16
4.8. Analytical Toolbox	17
4.9. Formal Analysis of the Integrated Circuits	20
4.10. Anomaly Detection in Vehicular ad/hoc Network	20
4.11. IoT Discovery and Classification	21
5. Possible Exploitation	23
References	24

1. SUMMARY

The cyber-threat landscape associated with IoT is diverse, developing rapidly and has enormous implications for the security of IoT enabled cyber physical systems. The integration of resource-constrained IoT devices with the global Internet makes many traditional security analysis techniques inapplicable. Therefore, novel approaches for formally analyzing hardware and protocols, generating test cases, intrusion and threat prevention, in-situ device and anomaly detection technologies are needed to overcome this issue. This whitepaper describes the IoT4CPS approach to enable safe and secure IoT-based applications for automated driving and for smart production, addressing strategic security assurance and security during operation:

- It shows the use of formal hardware property checks in order to provide security on basic building blocks of components at the network layer.
- Describes the formal verification of side-channel protected hardware implementation, which resulted in the first formally verified AES Substitution-box design that requires only two random bits for the initial sharing of its inputs and requires no online randomness to achieve first-order security in the probing model.
- Furthermore, hardware apps and the use of dynamically exchangeable runtime checkers as hardware apps, in which several functions were implemented, are described.
- Shows a threat modelling approach based on the STRIDE model with an extended template model tailored for the IoT4CPS project.
- Demonstrates automated security test generation. and specific penetration testing based on a threat model.
- Also, the human aspects in automated model checking of security protocols were addressed by symbolic model checking where the formal verification of human errors as well as usable symbolic model checking for engineers were evaluated.
- Outlines machine-learning techniques to detect anomalies at the log file, network communication and hardware level.
- A reliable IoT device classification and network discovery at runtime, to outline the network's structure and to reveal weak spots in today's networks, makes a further contribution to the comprehensive security approach of IoT4CPS.

2. CHALLENGE

The digitalization and increasing connectivity of (critical) cyber-physical objects enable the development of new applications but also leads to new safety & security related requirements in the design, testing, production and operation of these systems with resource constrained devices. This raises the need for novel approaches for formally analyzing hardware, protocols and system architecture as well as generating test cases. Besides these strategic security assurance aspects, also security during operation has to be accomplished by taking into account emerging threats.

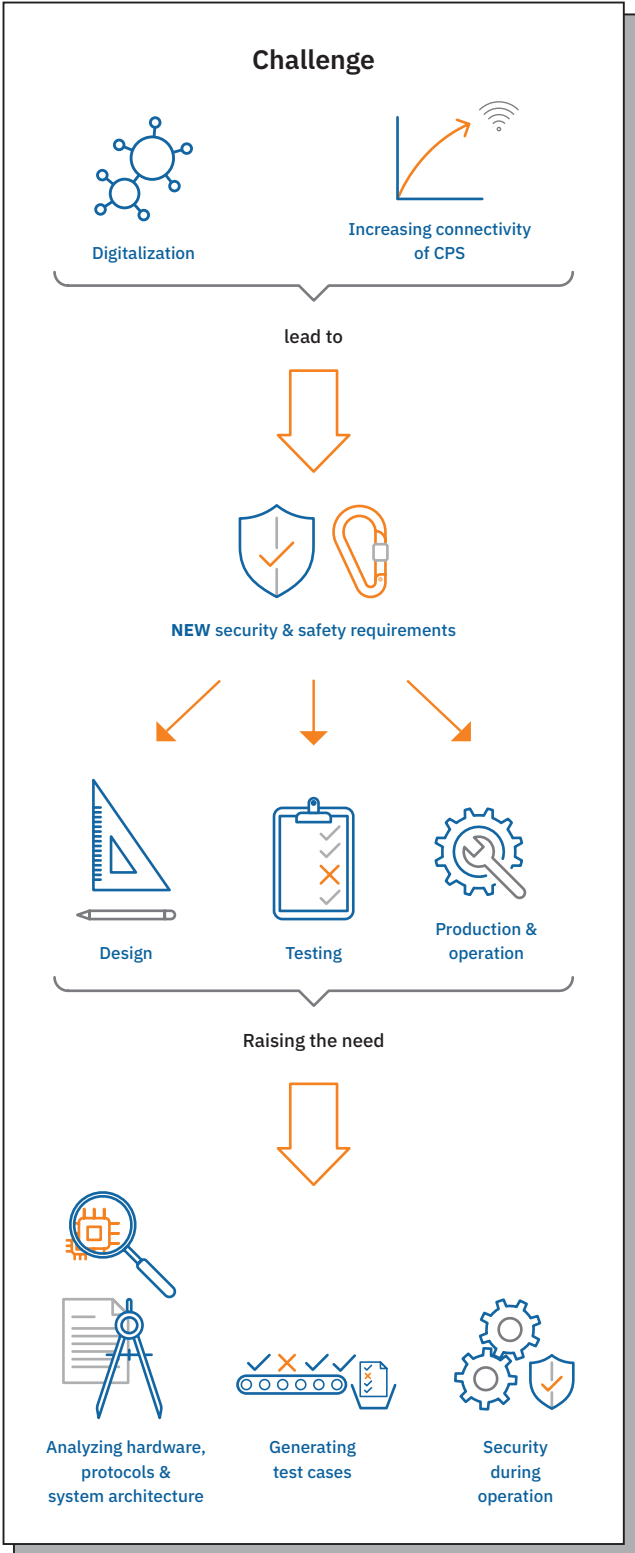


Figure 1: Challenges appearing with the digitalization and increasing connectivity of critical cyber physical systems.

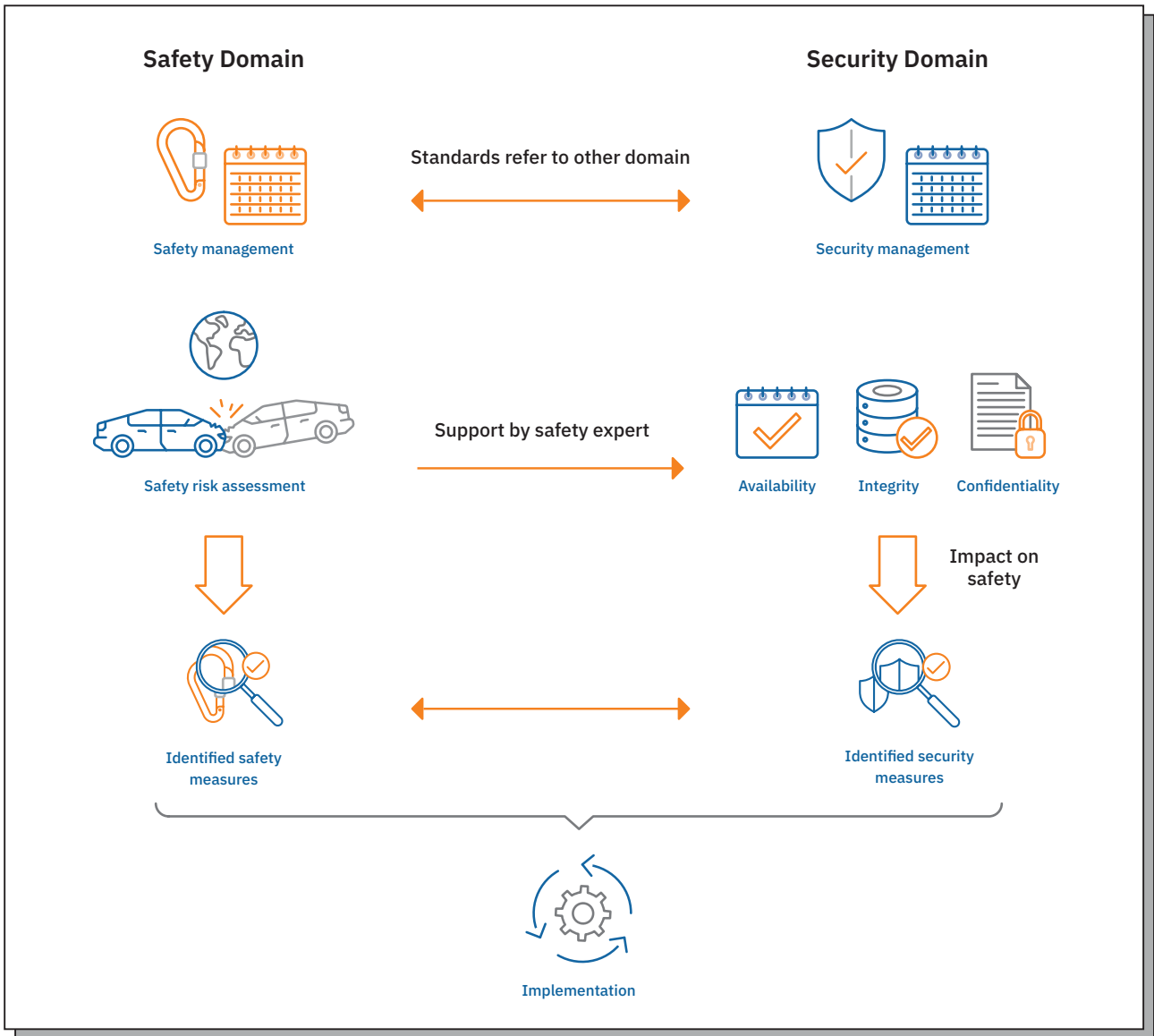


Figure 2: IEC TC65 WG20 “Industrial-process measurement, control and automation – Framework to bridge the requirements for safety and security”

An important point is also the interaction between newly developed security engineering processes and existing safety engineering processes. As example IEC TC65 WG20 “Industrial-process measurement, control and automation – Framework to bridge the requirements for safety and security” develops guidelines for the interaction between safety and security in the industrial domain.

2.1. Automated Driving

The importance of security and in particular its interdependencies with safety has been impressively demonstrated in July 2015 by a so called white hat attack, research-based hackings to discover vulnerabilities, which were applied via a remote attack on a Jeep. Black hat attacks are attacks triggered by cyber-criminals to cause security and safety issues respectively request ransom money. In 2018, black hat attacks have exceeded the incidents discovered by white hat hackers. Therefore, increasingly fast and continuously updated verification and analysis methods are needed to ensure security over the entire lifetime of a car (Figure 3).

2.2. Smart Production / I4.0

Industry 4.0 comes with the promise of increased efficiency and numerous new services and business models as a result of highly interconnected and thus more dynamic control and production systems. Sensors and numerous small computational components in industrial networks will become mainstream for every step of the industrial production process and will transform from local Operational Technology networks to (I)IoT connections and usage of 5G.

This in turn will allow connecting production to logistics, customers to products on the shop floor, and robots and workers to each other. Since industrial systems stay in the field for long periods (several years to decades), the security margins are expected to be reduced over time. Therefore, there is a need for updatability and resilience to recover some baseline functionality in case a system is compromised. Furthermore, several layers of hardware and software security and isolation will be required to keep security manageable since availability and safety are of high importance in industrial applications. The move towards a more consistent security approach for (I)IoT requires proper security assurance and standardized ways and procedures to evaluate the security of solutions.

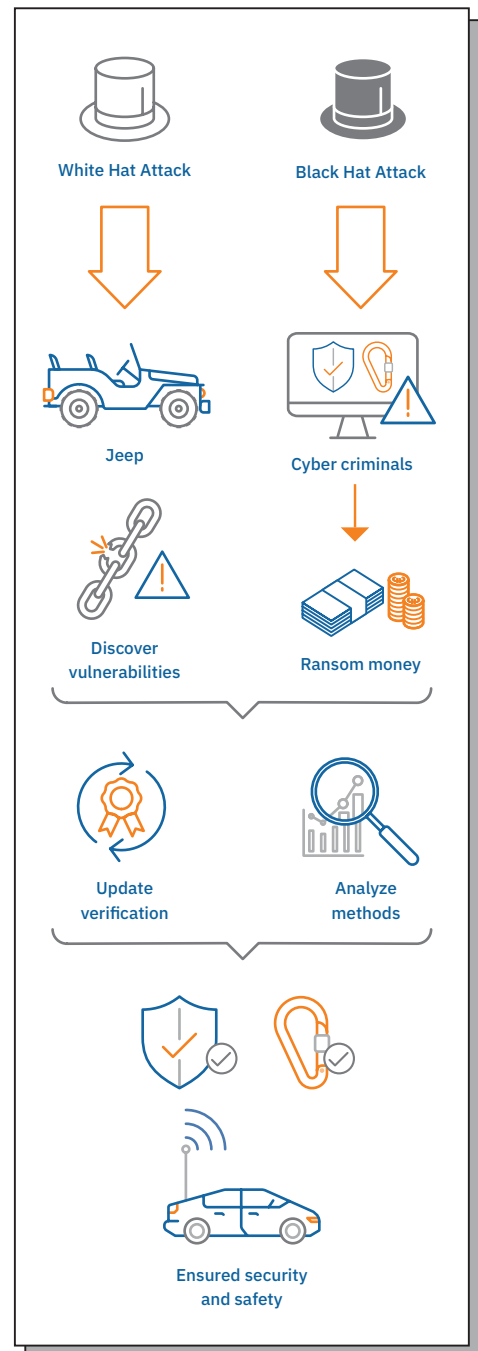


Figure 3: Fast and continuous updated verification and analysis methods to ensure security over the entire lifetime of a car is needed

¹ Greenberg, Andy. „Hackers remotely kill a jeep on the highway—with me in it.“ Wired, 21 July (2015)

² <https://62a.61a.myftpupload.com/wp-content/uploads/2018/12/Black-Hat-versus-White-Hat-Cyber-Attacks-Courtesy-Upstream-Security.png?time=1597236976>

3. CURRENT STATUS

Traditional verification and analysis techniques for hardware, software and networks cannot be applied directly in the two use case domains of automated driving and smart production, as it would not fully meet the security requirements of upcoming dynamic IoT setups. Therefore, IoT4CPS develops a full set of approaches to assure security by both formal and empiric methods to detect anomalies and uncover vulnerabilities at different system levels (Figure 4).

3.1. Protection Profile

While the recent cybersecurity guidebook for cyber-physical vehicle systems (SAE J3061) focuses on integrating cybersecurity in automotive processes, it falls short in addressing and enumerating threats and their mitigation strategies. An open discussion of these threats and mitigation strategies would greatly reduce the overall automotive security risk and a Common Criteria (CC) approach could be a tool to structure such a discussion.

3.2. Formal Verification of Side-Channel Protected Hardware Implementation

Side-channel and fault attacks are severe threats against cryptographic keys that are used in IoT devices to provide security. A central challenge when implementing countermeasures against these attacks is to verify that this implementation is correct and provides the desired level of protection. A standard approach is to prototype a design, to do actual measurements and analyses of the side-channel leakage. However, this is a very time consuming and incomplete approach, as testing is done for a specific attack setup and analysis technique. The recent progress uses formal methods to show security properties of hardware implementations of countermeasures like masking. Such approaches allow verifying the security of implementations at design time and providing precise security bounds. Formal verification is also an established method of analyzing communication protocols.

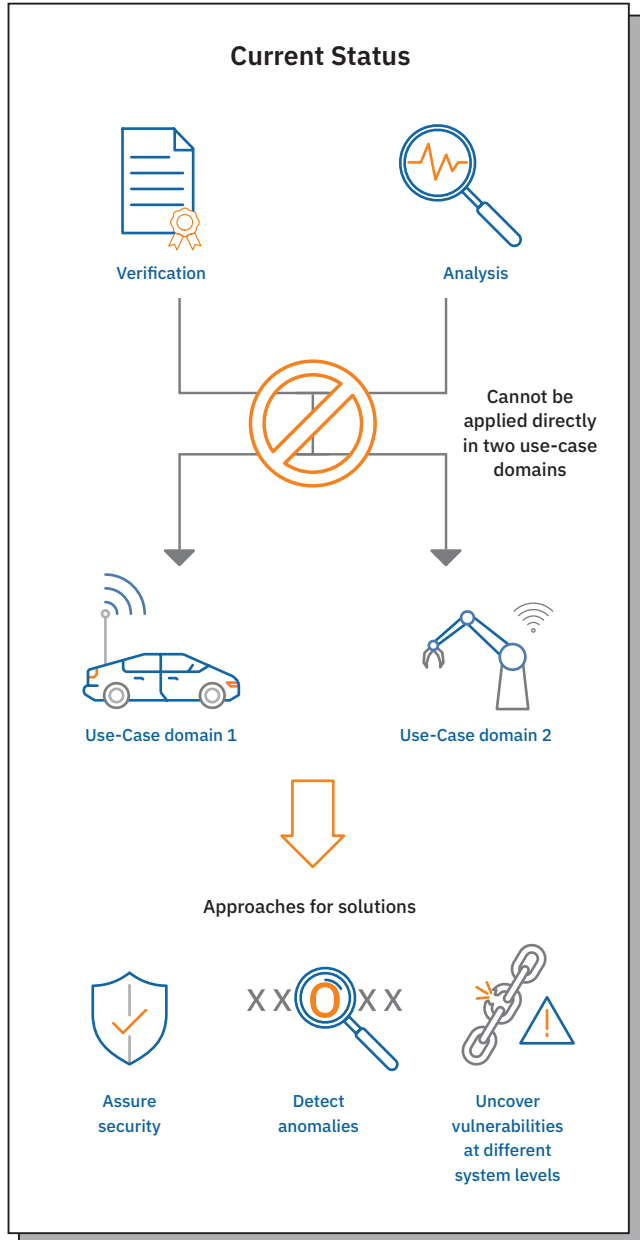


Figure 4: The IoT4CPS approach

3.3. Hardware Property Checks

The interconnectivity between several devices has raised security issues in the IoT, especially the network layer. The security issues in network layers include information stealing, communication channel jamming, spoofing, denial-of-service, etc. Traditionally, these security issues are addressed at the application layer, protocol layers, or system level. However, these techniques cannot ensure the trustworthiness of each component. For example, in the case of compromised hardware, i.e., a small piece of hardware that performs a security attack when it gets a trigger, known as Hardware Trojan (HT), these traditional techniques cannot guarantee the privacy of information.

3.4. Dynamically Exchangeable Runtime Checkers in HW

Software Apps are available for almost any modern mobile device such as Smartphones or Tablets. These small programs can individually enhance the default functionality of the device on demand. Apps can be purchased and installed via an App-Store that is maintained by a service provider. These well-known and widely accepted concepts of the mobile communication domain are increasingly introduced to other sectors such as Automotive, Industrial Control and Automation, where more stringent safety and security requirements need to be considered. Especially in systems where battery runtime is of less importance, reconfigurable hardware devices are integrated into many cases. FPGAs (Field Programmable Gate Arrays) offer great flexibility with respect to application openness, since they can be reconfigured on-the-fly during runtime. Latest FPGA products offer the possibility to even exchange only portions of the device while the rest of the system continues its operation without interference.

The concept of Hardware Apps extends standard Software Apps with hardware accelerators instantiated within reconfigurable FPGA devices. Consequently, this new kind of App consists of two parts, both available via App-Store:

- Software App
- Hardware configuration (FPGA bit stream suitable for partial reconfiguration) Hardware Apps are applicable wherever spare FPGA resources are available in order to accelerate SW-Apps or simply to offload the CPU.

3.5. Threat Modelling

There are different tools and techniques of threat modelling available (*Figure 5*), covering different aspects, such as data, data flow, application and assets, orientated on the risk or impact, respectively different application areas like software engineering or system architectures. Process Flow Diagrams are particularly useful for identifying threats of applications or IT architectures and can help software developers, software testers, as well as system architects and cyber security experts to mitigate threats during the development process.

Data Flow Diagrams, which are created from an attacker's perspective, provide operational threat models of the infrastructure, which provide a better view of the entire attack surface and are typically used within the DevOps lifecycle.

3.6. Testing

Testing is one of the most widely practiced fault-finding approaches. In testing, the system is executed with test input and the observed result (i.e., output and system behavior) is analyzed and compared to the expected result. Testing has been adopted for finding security faults in the most common form of penetration testing and fuzzing. In penetration testing, an expert performs a simulated attack of the system under test to reveal weaknesses and security loopholes. Since pen testing is a manual approach and the simulated attacks are „handcrafted“, it is limited by time and human resources.

In contrast, fuzz testing is a fully automated approach. Invalid input is generated and fed to the system under test to trigger crashes due to implementation weaknesses that can be exploited. Fuzzing has gained considerable attention since the increasing availability of computing power has enabled this testing approach to reveal many security issues in popular applications, operating systems, libraries, device drivers etc. The inherent limitation of fuzzing is that the type of faults that can be detected is mostly restricted to crashes. Many other critical security issues such as information leakage, unauthorized access, data corruption are not automatically detectable. Furthermore, with IoT/IIoT the complexity of the system under test can rapidly increase with the consequence that fuzzing may get stuck at the interface layer and the approach fails to test the components located in the deeper layers of the IoT system.

3.7. Human Aspects

In order to obtain secure cryptographic protocols for CPS, human factors must be taken into account. Ultimately, it is the human who is responsible to design, deploy, use, and maintain these systems. Attackers commonly target humans operating protocols instead of machines since the former are more error-prone. Therefore, humans represent a weak point which, if left unattended, can have a negative impact on all other parts of the system. However, it is challenging for CPS protocol designers to consider possible human errors in different environments and understand how they play together with the broad threat landscape surrounding the IoT ecosystem. In general, the security analysis of IoT protocols is a hard task, as vendors often react quickly on market demands and frequent code changes are required. Moreover, many different types of adversaries must be considered, depending on the deployment environment.

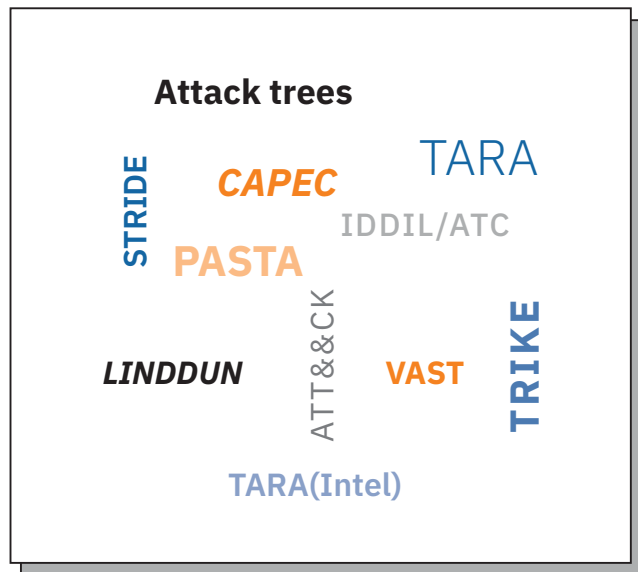


Figure 5: Threat Modelling Techniques

Symbolic model checking is a useful approach for automatically detecting security flaws in cryptographic protocols, which has been recently used to uncover security vulnerabilities in several deployed security protocols (e.g., MQTT and CoAP, the vehicular networking (V2X) revocation protocol). Albeit available model checking tools are promising but do not sufficiently take human factors into account.

First, the possibility of human error is not reflected in state-of-the-art symbolic model checking tools by default. Second, usability challenges of these tools are among the main reasons why they have not yet gained wide acceptance in industrial use cases and are currently only applied in academic circles.

3.8. Analytical Toolbox

An important aspect to operate cyber physical system in a secure and safe manner is to have a constant overview of the current status and system activities. If a deviation from the expected behavior is detected, this anomaly may be an indication of a security breach caused by an attack to the system.

Security experts are often unaware of the attack patterns and intrusions. Thus, defining patterns in data for algorithmic detection of attacks and intrusions is not feasible. Even if researchers and security experts identify some of these patterns, the attackers will move on to other or improved approaches and tools. For this reason, companies are seeking for software tools to support them in their task to monitor and understand the risk and security threats within their systems.

One approach to this problem is to use machine learning for anomaly detection. It is similar to outlier detection in statistics. The general approach is to train models that represent the typical system behaviour and then identify deviations from it. These deviations are the anomalies. It is of importance to note that this approach does not assume any knowledge about the attacks, tools for attacking, or about the attackers. It is a semi-supervised machine learning approach.

3.9. Formal Analysis of the Integrated Circuits

Different mathematical modelling and formal verification-based vulnerability analysis techniques of integrated circuits have been published and proposed. The analysis of security vulnerability of Integrated Circuits using conventional design-time validation and verification techniques (like simulations, emulations, etc.) is generally a computationally intensive task and incomplete by nature, especially under limited resources and time constraints.

3.10. Anomaly Detection in Vehicular Ad-Hoc Network

Vehicular ad-hoc network is emerging as the prominent communication framework for autonomous vehicles. However, due to complex, dynamic, and heterogeneous communication network topologies, protocols, and devices, these networks are vulnerable to several security threats, i.e., information leakage, denial-of-service. Therefore, IoT4CPS developed a novel methodology that leverages the statistical modelling of communication patterns (i.e., Hurst Exponent) to generate a single parameter-based unique signature for

a particular type of communicating command. These signatures are then compared with the runtime communication pattern modelling to identify anomalous communication behavior.

3.11. IoT Discovery and Classification

Based on the cyber kill chain, which describes the steps an external attacker performs to penetrate a system, network reconnaissance both external and internal reconnaissance can be the basis for lateral movement within a network. There are different tools available, which support administrators to perform a network scan or network audit. With the introduction of IoT technology and their topological peculiarities, in particular a concise and up-to-date overview of the network topology is increasingly difficult to achieve. The huge number of intertwining devices and protocols is making it increasingly tedious to create a reliable picture of the network. That is especially the case in cyber-physical environments, such as smart factories.

4. RESULTS

4.1. Automotive Ethernet Protection Profile

Ethernet is currently emerging as an increasingly important vehicular network bus due to the fact that two cost-effective physical layers are available for automotive Ethernet (IEEE 802.3 100BASE-T1 and 1000BASE-T1) and that the real-time and robustness properties have been improved by the IEEE 802.1 AVB (Audio/Video-Bridging) and IEEE 802.1 TSN (Time Sensitive Networking) standards.

IoT4CPS has defined an experimental version of a protection profile for automotive networks based on Ethernet. This protection profile specifies the Target of Evaluation (TOE) and outlines the security problem, security objectives, as well as formal Security Functional Requirements (SFRs) and different Evaluation Assurance Levels (EALs).

The defined TOE consists of Deterministic Ethernet switches and CAN gateways and the wire harness that connects the switches to each other as well as the wire harness connecting the user to the TOE. The TOE is completely located inside a car and the considered in-vehicle network architectures cover multiple communication standards, including, Audio-Video Bridging, Time-Sensitive Networking, and Time-Triggered Ethernet in combination with a BroadR-Reach® physical layer. A vendor implementing this protection profile will decide on the appropriate EAL case by case.

4.2. Formal Verification of Side-Channel Protected Hardware Implementation

Side-channel attacks in general and in particular power-analysis attacks are a severe threat to implementations of cryptographic algorithms. One approach towards counteracting power analysis attacks is to mask the intermediate results of a cryptographic algorithm. Many approaches to efficiently mask implementations of cryptographic algorithms have been proposed in the last two decades. However, many attacks have also been found. Consequently, there is a significant focus on formally verifying masked implementations. IoT4CPS has demonstrated that first-order masking is theoretically possible with only two random one-bit masks. As a first practical contribution, we designed a masked AND gate that allows reusing randomness from its inputs safely [1].

Based on our findings, we introduce a simple rule-based system. These rules can be encoded in Simultaneous multithreading-2 statements and they are then used to automatically check whether the masking approach is directly applicable to an unprotected implementation or if modifications (mask changes) are required. Upon acceptance, our tool synthesizes a securely masked implementation for a given set of additional constraints like the used mask encoding. We then show how our approach can be applied to larger implementations and demonstrate its feasibility and its impact on a full AES-128 encryption-only implementation. With our approach, we successfully designed the first formally verified AES Substitution-box design that requires only two random bits for the initial sharing of its inputs and requires no online randomness to achieve first-order security in the probing model [2]. Even when going for a full AES implementation, the randomness requirements do not increase further.

4.3. Hardware Property Checks

IoT4CPS developed a run-time methodology for Hardware Trojans detection that employs a multi-parameter statistical traffic modelling of the communication channel in a given System-on-Chip (SoC). The proposed methodology leverages the statistical traffic modelling of communication channels in the SoC to sniff the possible anomalies in 3rd Party Intellectual Property (3PIP) units [3]. The proposed methodology consists of the following three key steps.

Statistical modelling of communication behavior: To extract the golden communication behavior that can be used during the run-time for HT detection. We assume that in an SoC at least one of the IPs is trusted, and by analyzing the communication of the trusted IP, we can extract the requires golden behavior. Therefore, we propose to statistically model the normal communication traffic of the trusted IP, during the design phase of the SoC. This behavior is extracted by obtaining a statistical traffic model, which is characterized based on the following observations:

- How often are the packets injected in the communication channel?
- On average, how far does each packet travel?
- What portion of the total traffic has been injected in the communication channel by each module?

Hurst Exponent: We propose to choose the Hurst exponent because of the following reasons:

- It is very sensitive to the distribution of the time series.
- It requires very few observations to estimate the Hurst exponent. For example, the minimum observations required for Hurst exponent is approximately 240.

Hardware Modules Designing: The runtime verification of the Property Specification Language (PSL) assertion based on statistical parameters requires hardware modules that compute the above-mentioned statistical parameters. Therefore, during the design time, the designer designs these hardware modules and integrates them into critical communication channels. Moreover, the designer also translates the statistical model of communication behavior into their corresponding PSL assertions. These assertions are embedded into the RTL description of SoC along with the traffic monitoring units.

Run-time monitoring: During the run-time or testing time, the values of the statistical parameters calculated from the corresponding hardware modules are used to verify the associated PSL assertions. If one of the assertions fails the verification, then the communication channel is considered as intruded. Note, the traffic modelling is done under the premise that at least one of the IPs is trusted.

4.4. Dynamically Exchangeable Runtime Checkers in HW

True Random Number Generators (TRNG) are essential security building blocks that are used to generate random numbers with high entropy. Often, FPGAs (Field Programmable Gate Arrays) are used to host cryptographic functions when random numbers are needed, and many different mechanisms and physical effects can be exploited to realize TRNGs in FPGA reconfigurable logic. Especially for FPGA implementations,

it is difficult to estimate respectively measure the entropy for a TRNG because it may depend on the actual final placement within the available FPGA fabric. Even device variations can have a significant impact on the quality of extracted random numbers.

In IoT4CPS Dynamic Partial Reconfiguration is used to build an FPGA Design where individual checker modules are realized as HW-Apps that operate like a firewall. These checkers are located next to the interfaces of critical modules (e.g. encryption engine or random number generator) and observe data transfers and in case a malicious activity is detected, an alarm is triggered via an interrupt line. For example, HW_App Checkers can provide functions to assess the randomness of a bit sequence generated by a random number generator (RNG). This bit sequence is a part of encrypted data packets. The bit sequence to be tested will be extracted from input data under consideration of the input data format and protocol. The functionality for the extraction of RNG_data is common for all Checkers and is implemented in a separate sub-module. The functions for checking the randomness of data are taken from the NIST test suite, using generic parameters that can be configured and are retrieved for statistical evaluation of randomness.

4.5. Threat Modelling

Due to the constantly increasing number of IoT components with different characteristics, it is important to use a modelling tool that is as extensible as possible and that allows the specific modelling of data flow and processes. In IoT4CPS, various threats have been identified already during the specification of the use cases. These were also specifically assigned to individual processes or interacting components. The implemented MS Threat Modelling Tool enables the modification of the underlying modelling template and therefore it is possible to address individual threats and to specify the IoT4CPS components and data flows used in the use cases according to the underlying terminology (STRIDE) [4]. Based on an existing model for the Device Connect Use case, the advanced model yield to a final list of 358 threats which then were reviewed in cooperation with the developers of this platform in order to have full insight if the respective threats are applicable or not. Thus, out of the 358 generated threats, 246 threats were identified which resulted in the same mitigation even though the threats might occur on different places of the system architecture, 74 threats were not applicable or have been already addressed by the developers, resulting in 38 new threats that have to be addressed.

4.6. Testing

Automated Security Test Generation

Effective security testing has a sweet spot between manual penetration testing, which is time- and resource-consuming, and automated fuzzing, which is restricted to detect crashes and has limitations when testing complex IoT/IIoT systems.

An architecture and tool support for **automated security test case generation** has been developed. It employs a test adapter to bridge the gap between a domain- and technology-agnostic test generator (e.g., a random input fuzzer) and the functional and non-functional aspects of the system under test. The adapter is able to communicate with the system under test over IoT/IIoT protocols. A prototype implementation of

the adapter is provided for the popular MQTT publish-subscribe messaging protocol. On the side of the test case generator, the adapter exposes the system’s functionality in form of high-level parameterized commands. These commands also check for expected responses of the system under test. For example, that after performing a valid connect with authentication, the connection is actually established with the correct user. This extends the capabilities of the generated test cases beyond simple crash detection used in fuzzing. Furthermore, it also allows simulating complex interaction scenarios such as concurrent access to a server by multiple clients.

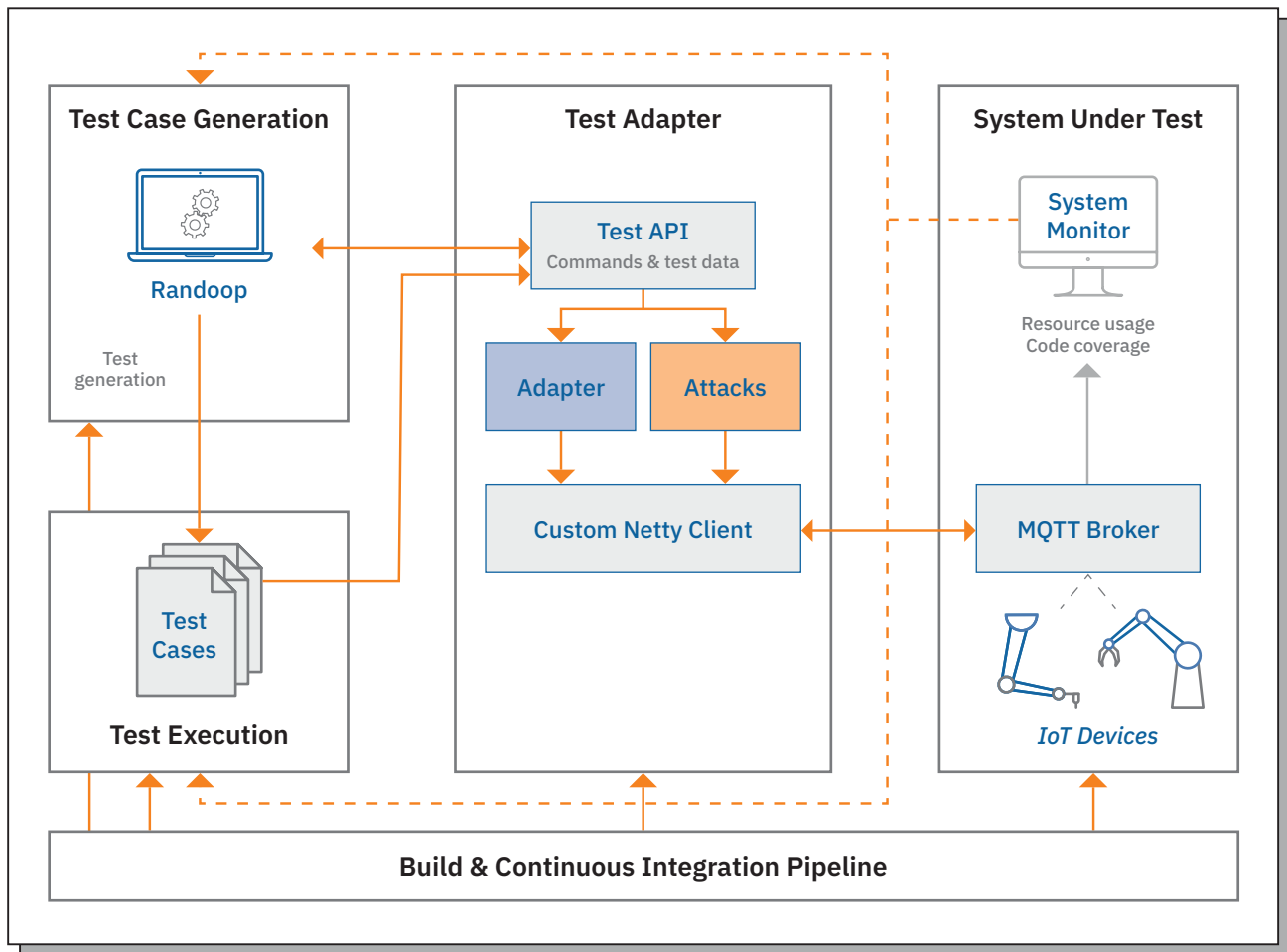


Figure 6: Core Components for automated security test case generation

The generated test cases can represent valid and invalid interaction scenarios. In the case of invalid scenarios, when executed, the tests verify the robustness of the system under test and they check that it responds gracefully, e.g., with an error message. The same system behavior is expected for illegal scenarios containing malicious interactions that resemble security attacks. A secure and robust system prohibits such malicious interactions. It responds with an appropriate error message and continues to operate correctly without negative side effects. In our approach, attacks are implemented as an extension to the adapter, which allows configuring test generation to create test cases consisting of valid interactions interleaved with invalid and malicious interactions. In that way, even complex interaction scenarios where an attack is performed after triggering a system error can be automatically generated.

A demonstrator for automated security testing has been built for the use case of secure data exchange for distributed connected devices in an untrusted environment. The approach is shown for data hubs using the MQTT messaging protocol as system under test. A total of 21 automated attacks have been implemented to generate automated tests that target potential threats. These attacks have been aligned with the above mentioned threat model for the Device Connect Use case [5] [6] [7].

A **penetration test catalogue** was also created for automated security test case generation. The catalogue describes test cases, their prerequisites and test steps in order to help cyber security experts, system administrators, software developers and software testers not only to test their products in regard to cyber security but also to increase awareness and communication during the development and testing processes. In addition, this penetration test catalogue can assist the cyber security experts in conducting a penetration, which is the process of testing computer systems as well as human resources (social engineering) to identify security threats and possible vulnerabilities.

4.7. Human Aspects

We discuss the first attempt to integrate human behaviour into automated model checking tools and its suitability for verifying IoT protocols. Basin et al.³ evaluate different authentication protocols and discuss their resistance to human error when assuming infallible, fallible, rule-based, and skilled humans. They create heuristics which can be applied by secure authentication protocols in general to prevent human error. **Our evaluation results suggest that these heuristics also apply to IoT protocols and should be taken into account when designing authentication protocols for CPSs.** In particular, Basin et al.³ suggest to enforce crucial human operations such as carrying out certain checks as far as possible in order to minimize the space for skipping these steps. For instance, the human can be forced to enter a code instead of being requested to compare two codes which can be skipped by fallible humans. Denzler⁴ introduces a tool that automatically generates all set of possible errors for a specific protocol. **It can be applied in the realm of CPS protocols since it offers possibilities to model heterogeneous environments and human knowledge bases.**

Novel approaches to overcome usability challenges of automated model checking tools and the evaluation of their applicability for IoT protocols in comparison to the original tools are presented in Section 7.4 in Deliverable 4.2.⁵ Kobeissi et al.⁶ introduce a publicly available online tool called Noise Explorer that automatically formally verifies protocols created with the Noise Framework.

The results of our analysis suggest that since the tool Noise explorer can only be used for handshake protocols, its applicability for model checking protocols of CPS is limited. However, the tool is a step in the right direction to make automated model checking tools more usable. In addition to Noise Explorer, Kobeissi et al.⁶ present a novel software called Verifpal for verifying security protocols. It aims to make automated

³ D. Basin, S. Radomirovic and L. Schmid, "Modeling human errors in security protocols," 2016 IEEE 29th Computer Security Foundations Symposium (CSF), 2016

⁴ A. Denzler, "Automatic Analysis of Communication Protocols with Human Errors," 2016

⁵ https://iot4cps.at/wp-content/uploads/2019/07/IoT4CPS_D4.2_V11-final_formatok.pdf

⁶ Kobeissi, "Verifpal: Cryptographic Protocol Analysis for Students and Engineers," Cryptology ePrint Archive, Report 2019/971, 2019

model checking more intuitive and usable for engineers. **The tool Verifpal offers multiple usability enhancements in comparison to traditional automated model checking tools and can be applied for user-friendly modeling of CPS protocols.**

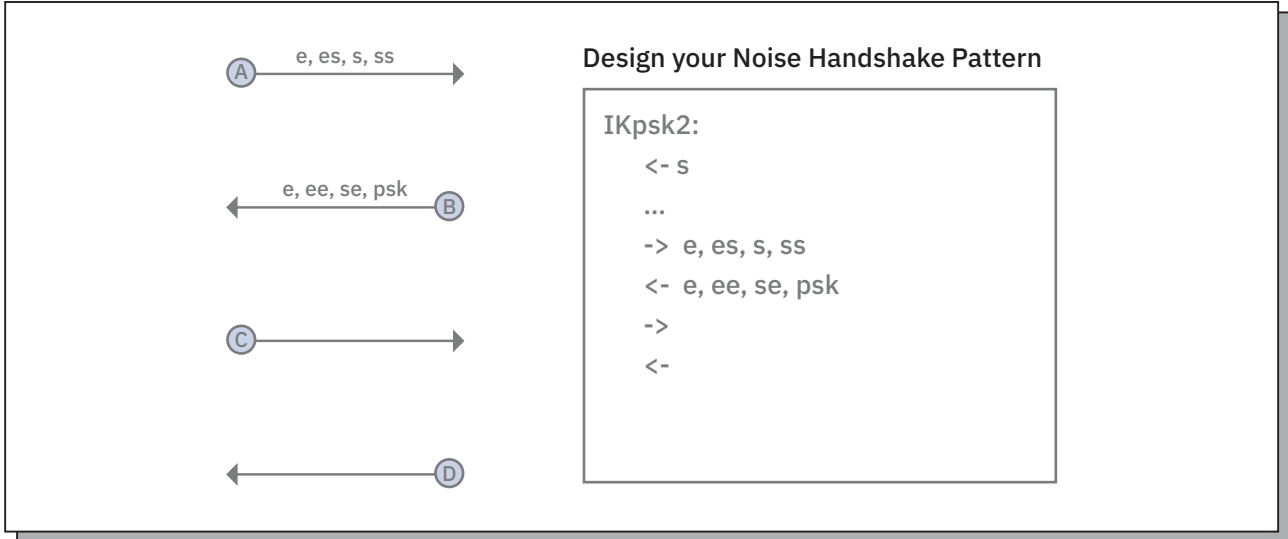


Figure 7: Example of a simple Noise protocol sequence flow (left) and specification of this protocol in the Noise Explorer input language (right)

4.8. Analytical Toolbox

To ensure operational security on cyber-physical systems, IoT4CPS developed a set of tools for the identification of anomalies based on different machine learning and correlation techniques. To cover a wide spectrum of possible risks to CPS, anomaly detection is applied on three disjunct levels, all with different and independent data sources:

- **Network traffic:** The data source is mostly TCP/IP information. This includes the IP addresses, TCP connections, size of packets, etc. For networks of cyber physical systems, this network communication data may be very homogenous. For example, data is sent every day at the same time from a cyber physical system.
- **Operating system:** The data source is comprised mostly of (textual) system logs and application events. This includes information about logins, executed programs, file system operations and so on.
- **Hardware:** The power consumption of particular hardware components together with physical measurements at the hardware level is the data pool for this type of analysis. An example of a data source is the power consumption of the device recorded by some sensors.

On the network level, we use a semi-supervised machine learning approach to detect anomalies [8]. In contrast, supervised approaches require training and testing data to be from the same probability distribution, i.e. it is based on the assumption that the testing set does not contain different behavior from that of the training set. This assumption is however not fulfilled in real world applications, as novel attack types are expected to occur in the future, as is the case also in the used data set.

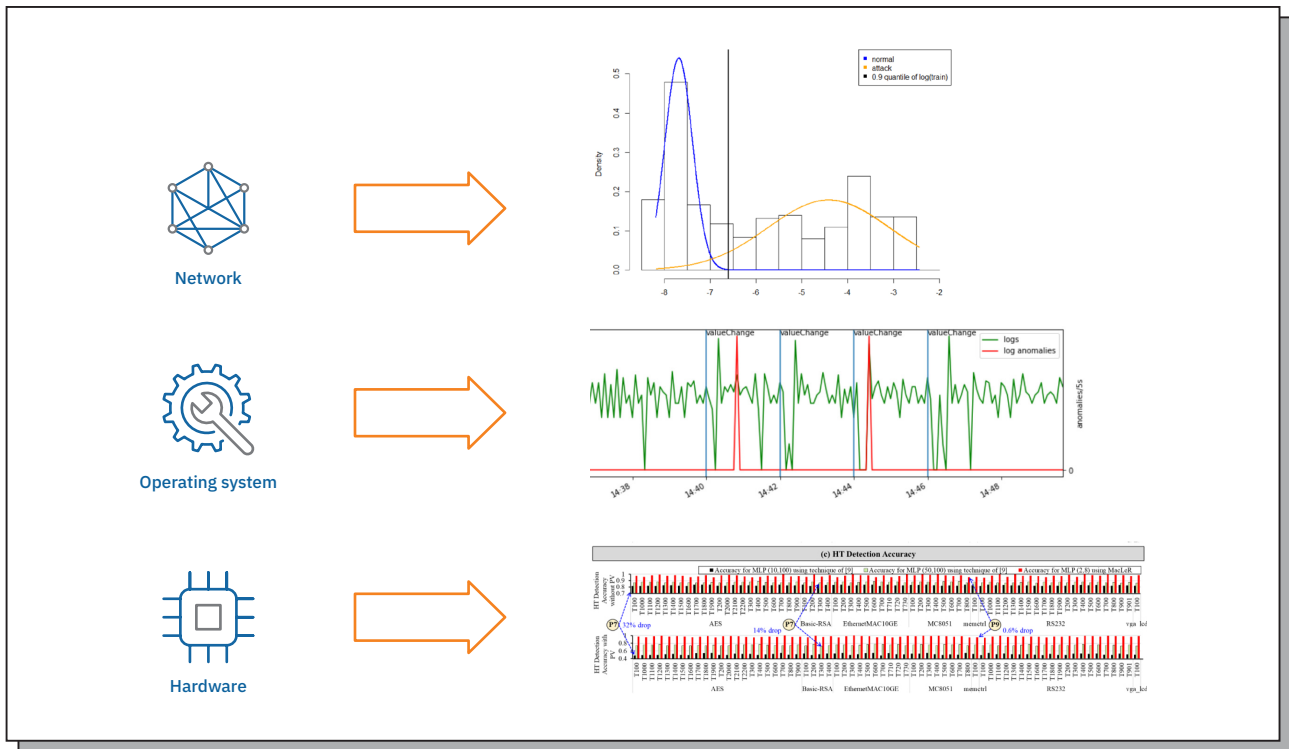


Figure 8: Anomaly detection performed on the network, system and hardware level of a cyber physical system

Another common issue in real world applications is either highly imbalanced training data set with only small proportion of attacks or alternatively only normal observations are available for training.

Thus, we use an anomaly detection technique, which will learn the representation of only normal samples and classify as anomaly each observation which is different from the learnt patterns. The advantage is that it does not suffer from insufficient number of malicious samples for training and novelties in future data are also not a problem as all attacks are considered anomalies. The assumption for this approach to work is that the behavior of legitimate users does not change in future.

Specifically, we train an undercomplete autoencoder on the samples consisting of normal traffic only and afterwards to distinguish anomalous observations as those with reconstruction error higher than the selected threshold. This means that the deep learning like structure of the autoencoder is used to learn the representation of normal samples and when the reconstruction error will be too high, the sample is considered to come from different probability distribution and will thus be labeled as an anomaly.

On the system level, the anomaly detection focuses on processing log data from operating systems and applications. The Automatic Event Correlation for Incident Detection (\mathcal{AECID}) software system [9][10] is used as a basis. Since \mathcal{AECID} uses self-learning and white-listing approaches, it is ideal to process logs produced by legacy systems and by appliances with small market shares (like those largely employed in cyber-physical systems). Furthermore, due to its decentralized architecture with the lightweight AMiner component, that can be used on systems with only minimal processing power and memory resources, \mathcal{AECID} is ideal in an IoT environment.

Consider an attacker who physically influences a sensor value and thus cause local damage. In this exemplary case the $\mathcal{A}ECID$ system would identify the following two anomalies at two different points in time:

- 14:41: An anomaly that is more likely to be false-positive is detected. The anomaly is triggered because the distribution of certain discrete values in the logs changed. This is due to the fact, that the changes produced a high number of log lines and thus the distribution change occurred.
- 14:44: The actual sensor value change is detected. The anomaly contained the information that the previous normal distribution of the value has changed to an unknown, unassignable distribution. This is understandable since at this point in time the sensor value had already changed three times. The second anomaly indicates that the attack was successfully detected. The reason the anomaly was not detected immediately after the first change or immediately after each change is because a certain number of values are necessary for a meaningful statistical test. For this reason, detection is delayed until sufficient data is collected.

Finally, anomaly detection on hardware is needed to ensure operational security in the case of a compromised hardware, i.e. a small piece of hardware that performs a security attack when it gets a trigger (known as hardware trojan). The previously discussed approaches on network and system level cannot guarantee the privacy of information for such attacks.

The main focus here is to provide security of the basic building blocks of a system-on-chip which consists of several trusted and untrusted 3rd party intellectual properties. Therefore, we propose a novel methodology, called MacLeR [11], to design a machine learning based run-time hardware trojan detection technique that exploits the fine-grained power profiling of the microprocessor (see . MacLeR employs the following analysis and methods:

- 1) To obtain the fine-grained power profiles, we measure the individual power of each pipeline stage with respect to a particular instruction. Such choice is made as the impact of hardware trojans on fine-grained power profiles is relatively better noticeable as compared to the overall power.
- 2) To reduce the complexity and detection time, we use an off-chip monitor that collects analog power profiles and converts them into the digital domain. These power profiles are then used first for training a machine learning model at design time, and afterward use this model at run time for detecting hardware trojans.
- 3) To extract the fine-grained power profiles of a microprocessor during the runtime requires multiple power ports. Therefore, to reduce the number of power ports, we use a single power-port current acquisition block and accordingly measure the current in a time-division multiplexing manner.

As depicted in the lower part the experimental analysis shows that in the case hardware trojan benchmarks for MC8051, the trained model provides approximately 98% hardware trojan detection accuracy, and with a very small number of false positives and false negatives. However, for other hardware trojan benchmarks, MacLeR still provides approximately 90% HT detection accuracy.

4.9. Formal Analysis of Integrated Circuits

In order to address the completeness, feasibility and coverage issues in formal verification of ICs, IoT4CPS developed novel framework for formal analysis of security vulnerabilities called ForASec [12]. It performs a comprehensive security analysis providing the complete 100% coverage of parametric behavior (i.e., leakage power, dynamic power and propagation delay) and process variations. The proposed ForASec consists of the following key features (see an overview and design flow in):

Mathematical modeling of gates for Formal Analysis Tools: Typically, hardware Trojans (HTs) have direct or indirect impact side-channel behaviour of ICs. Therefore, to encompass this behavior of HTs, we propose the comprehensive mathematical models of basic logic gates, i.e., NOR, NAND and NOT, w.r.t the behavior of side-channel parameters while considering the process variations effects on different technology parameters, e.g., switch on resistance, oxide capacitance, carrier mobilities, gate capacitance, drain capacitance and source capacitance.

A **model checking-based analysis methodology** to ensure the complete coverage of security vulnerability analysis, against the multiple intrusions at different locations, for all gates, nodes and with respect to all input patterns.

A **verification/analysis algorithm** to address the state-space explosion problem for large-sized circuits, while considering the uncertain behavior of side-channel parameters.

To evaluate the effectiveness of ForASec, we implement certain key basic sequential circuits and all the ISCAS89 benchmark in the presence of trust-hub Trojan benchmarks. The experimental results demonstrate that ForASec is able to correctly identify the most vulnerable node(s) and the minimum-possible size of Stealthy Hardware Trojans (SHTs) that can be detected while analyzing the leakage power. Moreover, it also provides approximately 11x to 16x speedup in analysis time compared to state-of-the-art model checking-based techniques.

4.10. Anomaly Detection in Vehicular Ad-Hoc Network

IoT4CPS developed a novel methodology that leverages the statistical modeling of communication patterns (i.e., Hurst Exponent) to generate a single parameter-based unique signature for a particular type of communicating command. These signatures are compared with the runtime communication pattern modeling to identify anomalous communication behavior. The proposed approach was successfully demonstrated on the VANET communicating with 802.11p for several datasets, i.e., RioBusses-v2018, VANETjamming2018, and VANETjamming2014. This is on-going with expected future improvements.

4.11. IoT Discovery and Classification

The implemented network reconnaissance procedure [13] consists of three main phases. Within the first phase, passive scanner modules observe the network and gather information about hosts in the network. The second phase consists of an analyzer module processing the results from the passive scanners and determining network ranges that will serve as input for subsequent active scanners, which form the third phase.

The iterative toolchain-based scanning approach enables the analysis of IoT protocols by adding additional scanning tools and analytics algorithms.

It supports IPv6 network scans using the neighbor discovery protocol (router advertisement

and neighbor advertisement) as well as Bluetooth and LoRa networks scans. *Figure 9* shows the dialog that pops up when the user initiates an IoT scan. Some basic information must be provided by the user before the discovery process can start.

Figure 10 shows the determined network graph generated from the data collected by the IoT discovery process displaying the identified node and edge objects of the underlying network topology connected via Ethernet, LoRa and Bluetooth.

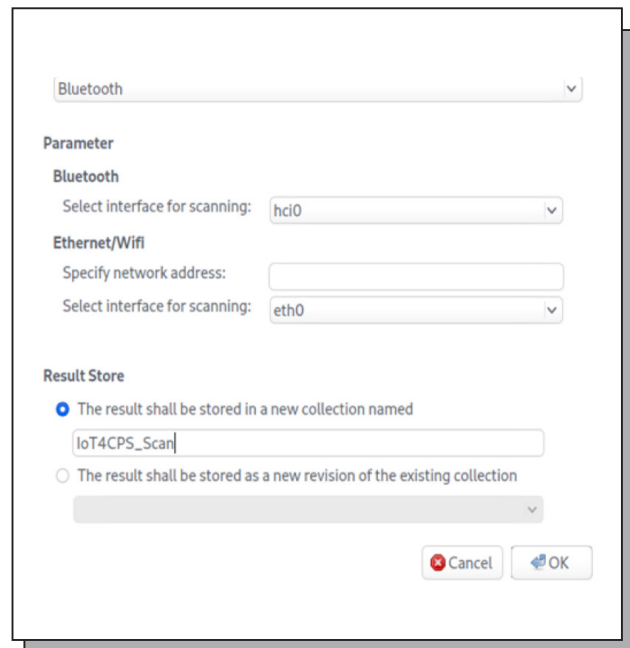


Figure 9: User Interface-IoT scan

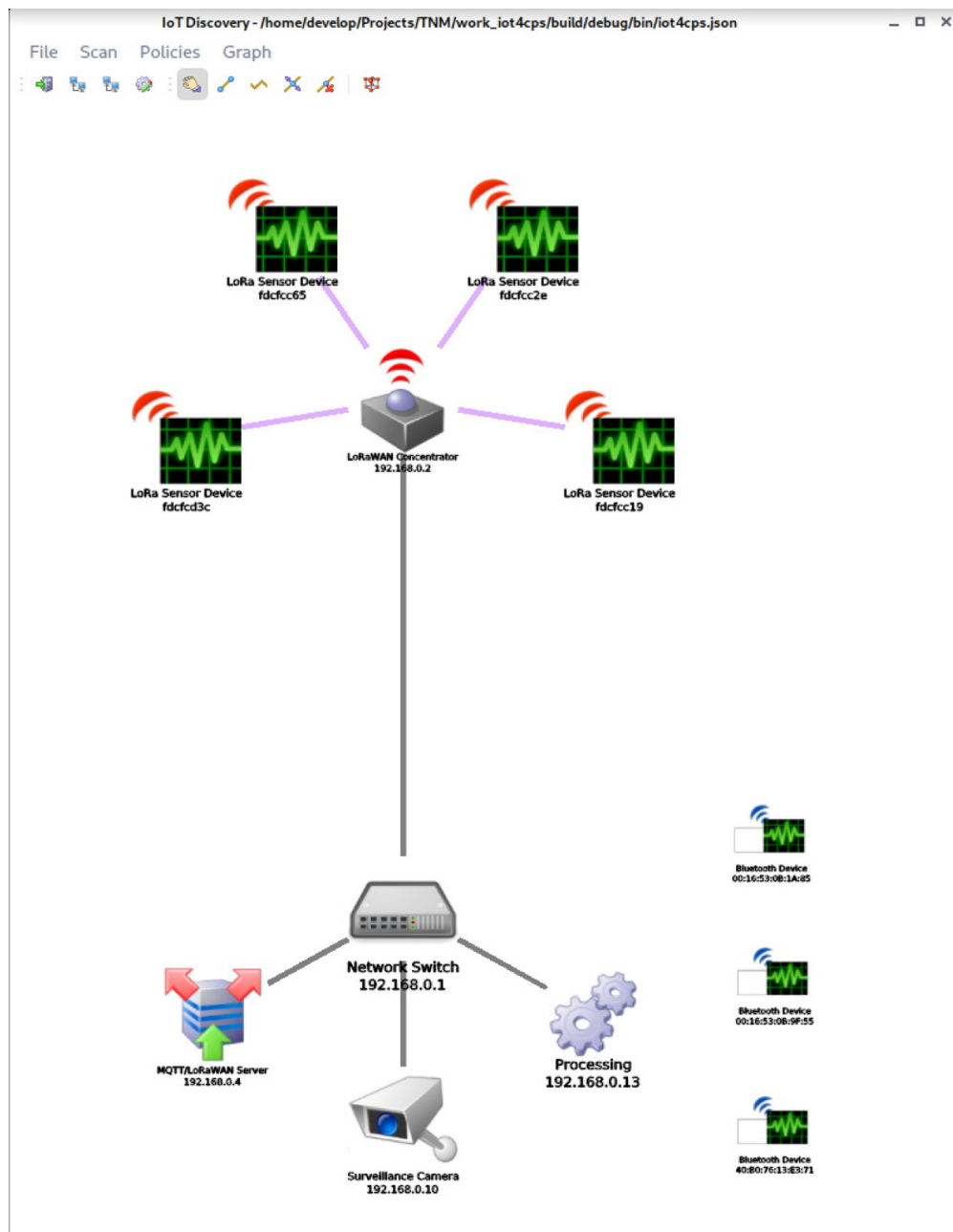


Figure 10: Resulting topology graph of the IoT scan

5. POSSIBLE EXPLOITATION

The comprehensive set of approaches assures security by both formal and empiric methods to detect anomalies and uncover vulnerabilities at different system levels. These are demonstrated and evaluated via IoT-based applications for automated driving and for smart production. Nevertheless, some of the described approaches are portable and can be used in other industrial domains as well. In certain cases, minor or medium adaptations are needed for the specific application area.

Security Approach	Automated Driving	Smart Production	Applicable to other Domains
Automotive Ethernet protection profile	● ● ●	⊗ ⊗ ⊗	⊗ ⊗ ⊗
Formal verification of side-channel protected hardware implementation	● ● ○	● ● ○	● ● ○
Hardware property checks	● ● ○	● ○ ○	● ○ ○
Dynamically exchangeable runtime checkers in HW	● ○ ○	● ○ ○	● ○ ○
Threat modelling	● ○ ○	● ● ●	● ○ ○
Testing	● ● ○	● ● ●	● ● ○
Human aspects	● ○ ○	● ○ ○	● ○ ○
Analytical toolbox	● ● ○	● ● ●	● ● ○
Formal analysis of integrated circuits	● ● ○	● ○ ○	● ○ ○
Anomaly detection in vehicular ad-hoc network	● ● ●	⊗ ⊗ ⊗	⊗ ⊗ ⊗
IoT discovery and classification	● ● ○	● ● ●	● ● ○

- ⊗ ⊗ ⊗ not applicable
- ○ ○ applicable with major adaptation effort
- ○ ○ applicable with medium adaptation effort
- ● ○ applicable with minor adaptation effort
- ● ● applicable without adaptation

REFERENCES

- [1] R. Bloem, H. Groß, R. Iusupov, M. Krenn und S. Mangard, „Sharing Independence & Relabeling: Efficient Formal Verification of Higher-Order Masking,“ 2018.
- [2] H. Groß, K. Stoenen, L. D. Meyer, M. Krenn und S. Mangard, „First-Order Masking with Only Two Random Bits,“ 2019.
- [3] F. Khalid, H. R. Syed und S. Muhammad, „SIMCom: Statistical Sniffing of Inter-Module Communications for Runtime Hardware Trojan Detection,“ *Journal of Microprocessors and Microsystems: Embedded Hardware Design*, 2020.
- [4] R. Ankele, S. Marksteiner, K. Nahrgang und H. Vallant, „Requirements and Recommendations for IoT/IIoT Models to automate Security Assurance through Penetration Testing, Thread Modeling and Security Analysis,“ in *ARES-WISI*, 2019.
- [5] S. Marksteiner, R. Ramler und H. Sochor, „Integrating Threat Modeling and Automated Test Case Generation into Industrialized Software Security Testing,“ in *Central European Cybersecurity Conference*, 2019.
- [6] H. Sochor, F. Ferrarotti und R. Ramler, „An Architecture for Automated Security Test Case Generation for MQTT Systems,“ in *Workshop on Cyber-Security and Functional Safety in Cyber-Physical Systems*, 2020.
- [7] H. Sochor, F. Ferrarotti und R. Ramler, „Automated Security Test Generation for MQTT Using Attack Patterns,“ in *ARES WISI*, 2020.
- [8] R. Svihrova und C. Lettner, „A Semi-Supervised Approach for Network Intrusion Detection,“ in *ARES-WISI*, 2020.
- [9] M. Wurzenberger, F. Skopik, G. Settanni und R. Fiedler, „AECID: A Self-learning Anomaly Detection Approach Based on Light-weight Log Parser Models,“ in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, SCITEPRESS digital library, 2018, pp. 386-397.
- [10] M. Wurzenberger, M. Landauer, F. Skopik und W. Kastner, „AECID-PG: A Tree-Based Log Parser Generator To Enable Log Analysis,“ in *IFIP/IEEE IM 2019 Workshop: 4th IEEE/IFIP International Workshop on Analytics for Network and Service Management*, Washington, 2019.
- [11] F. Khalid, S. R. Hasan, S. Zia, O. Hasan, F. Awwad und M. Shafique, „MacLeR: Machine Learning-based Run-Time Hardware Trojan Detection in Resource-Constrained IoT Edge Devices,“ in *IEEE TCAD*, 2020.
- [12] F. Khalid, I. H. A. Abbassi, S. Rehman, A. Kamboh, O. Hasan und M. Shafique, „ForASec: Formal Analysis of Security Vulnerabilities in Sequential Circuits,“ in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.
- [13] S. Marksteiner, B. Jandl-Scherf und H. Lernbeiß, „Automatically Determining a Network Reconnaissance Scope Using Passive Scanning Techniques,“ in *International Congress on Information and Communication Technology*, London, 2019.
- [14] F. Khalid, M. A. Hanif und M. Shafique, „Exploiting Vulnerabilities in Deep Neural Networks: Adversarial and Fault-Injection Attacks,“ in *Side-Channel Attacks, Detection & Defenses at International Conference on Cyber-Technologies and Cyber-Systems*, 2020.

© Copyright 2020, the Members of the IoT4CPS Consortium

For more information on this document or the IoT4CPS project, please contact:

Mario Drobits, AIT Austrian Institute of Technology, mario.drobits@ait.ac.at

Layout & Grafik

Nora Novak, goldmaedchen Grafikdesign

Legal Notices

The information in this document is subject to change without notice.

The Members of the IoT4CPS Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the IoT4CPS Consortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

The IoT4CPS project is partially funded by the „ICT of the Future“ Program of the FFG and the BMVIT.



Bundesministerium
Klimaschutz, Umwelt,
Energie, Mobilität,
Innovation und Technologie

