Gesellschaft für Informatik e.V. (GI) GI. (Hrsg.): INFORMATIK 2021,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021 1759

Supporting Security in Industrial Automation and Control Systems using Domain-Specific Modelling

Robert Altschaffel[1], Ivo Hempel[1], Oliver Keil[1], Josef Schindler[2], Martin Szemkus[3], Jana Dittmann[1], Matthias Lange[3], Karl Waedt[2] and Yongjian Ding[3]

Abstract: This paper explores how domain specific modelling can be used to support the identification of potential vulnerabilities and risks in Industrial Automation and Control Systems (IACS) to enhance security by enabling a mitigation of these vulnerabilities. This approach can be used to support already deployed IACS or to include Security-by-Design and Security Defence-in-Depth principles in the planning of future facilities. This paper explores the requirements for such a modelling approach including domain and security specific aspects. Three interlinked aspects of IACS which require different modelling approaches are identified leading to three distinct types of models: Infrastructure, cyber-process, and physical process. These three types are relevant for different attack vectors and to judge the potential impact of any attack. This paper shows examples for these three models and how these models can be used to identify vulnerabilities with the aim to close them.

Keywords: Industrial Automation and Control Systems, Security, Modelling, Standards

1 Introduction

Security is of increasing importance in the domain of Industrial Automation and Control Systems (IACS) as shown by the increasing number of recent attacks [Al20, AS19]. IACS are cyber-physical systems - computer systems which can directly affect the physical world by attached actuators. Hence, a threat to the security of an IACS often carries negative implications for the safety of the physical process or environments associated with the IACS. Hence, an increase of security in IACS is needed.

This paper aims to improve the security of already deployed or currently planned IACS by using domain specific modelling. This domain specific modelling allows to describe the IACS in question in a way that supports the identification of potential vulnerabilities

[1] Otto-von-Guericke-University, Department of Computer Science, Universitätsplatz 2, Magdeburg, 39106, robert.altschaffel@iti.cs.uni-magdeburg.de, ivo.hempel@ovgu.de, oliver.keil@ovgu.de, jana.dittmann@iti.cs.uni-magdeburg.de

[2] Framatome GmbH, ICETA-G Department, Paul-Gossen-Straße 100, 91052, Erlangen, Germany, josef.schindler@covalion.net, karl.waedt@framatome.de

[3] Hochschule Magdeburg-Stendal, Magdeburg, 39114, martin.szemkus@h2.de, mathias.lange@h2.de, yongjian.ding@h2.de

and risks.

To achieve this, the domain specific modelling must consider all the factors which might affect the overall security of the IACS in question. Section 2 provides necessary background information as well as the requirement definition in Section 2.3. Based on these requirements, an approach for security-aware domain specific modelling is presented in Section 3. Section 4 closes with an outlook on further research.

2 Requirement Definition and Related Background

This chapter describes relevant terms and technologies in the scope of modelling IACS. Based on this knowledge, we define requirements that must be met by our modelling approaches proposed later in this paper.

2.1 Components of IACS

According to [Al20] an IACS is defined as "A communication network of Actors, Sensors and Processing units geared towards controlling a physical process". Therefore, these components are essential for IACS and relevant for modelling in this scope:

- **Sensor**: “Collects information about the environment [...]” [Al20]
- **Actuator**: “Manipulates the environment […]” [Al20]
- **Processing Unit** (short **PU**; here **Programmable Logic Controller**): “Evaluates the data gathered by sensors and/or gives control signals to actors.“ [Al20]
- **Communication Wiring**: “The physical and logical carrier that facilitates communication between sensors, actors, and processing units.” [Al20]

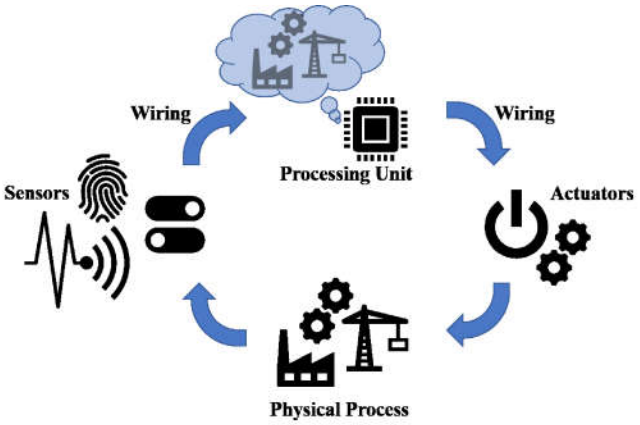

Fig. 1: Control loop with sensors, processing units and actuators in a physical process

Sensors measure a physical process while processing units compute how the physical process should be affected by the actuator implementing a control loop (see Fig. 1). This necessitates communication between sensors, processing units and actuators. This communication is handled by direct cable connections or more complex digital communication buses. Multiple control loops may be used to control complex physical processes. In addition to these control loops, other systems are usually included to provide an overview on the state of the entire physical process to an operator. Such a system is referred to as Supervisory Control And Data Acquisition (SCADA).

According to [Al20], the structure of these components follows a control hierarchy dictating the flow of communication within the network. A common and generalized way to describe such a control hierarchy is the Purdue Enterprise Reference Architecture (PERA; see [Wil92]). PERA consists of multiple levels of hierarchy. A brief overview on these levels can be found in [Ro11]:

- **Level 0**: Process sensors and actuators involved in the basic manufacturing process
- **Level 1**: Basic controllers (typically a PU) that direct and manipulate the manufacturing process
- **Level 2**: Area supervisory control applications and functions associated with the cell/area zone runtime supervision and operation
- **Level 3**: Site level plant-wide ICS functions

The higher levels provide business functions performed in an attached IT system. Although this summary references a manufacturing process, it is applicable to any other physical process as well. Within this structure, Level 0 components communicate with Level 1 components which in turn communicate with Level 2 components and so on. Hence, they form a control hierarchy which also acts as a communication hierarchy since

it dictates the flow of communication.

The hierarchy (and therefore the communication) might also be specified by further domain-specific restrictions. In this paper, IACS in the domain of Nuclear Power Plants (NPP) are taken as an example having high complexity and requirements for security. The IAEA provides an exemplary implementation for a Defence-in-Depth-Architecture (also Graded Approach, see [IA11]) based on different Security Levels (SL) and Security Zones which restrict the communication flow. The SLs are based on the impact a potential failure or compromise could have on the physical process. A summary of these SLs is provided in [Hi20]:

- **SL1**: systems vital to the facility (e.g. physical emergency protection)
- **SL2**: operational control systems which require high security
- **SL3**: supervision systems not required for operations
- **SL4**: technical data management systems (e.g. used for maintenance)
- **SL5**: business systems

While lower security levels should be able to send information to higher security levels, the information flow in the other direction should be highly restricted. Therefore, the levels are connected via access systems like firewalls or data diodes (see [Hi20]).

2.2 Attacks on IACS

Three different principal targets of typical attacks on IACS were identified in [St14]:

- communication stack of the deployed devices
- lack of authentication requirements within the hardware
- security problems in the respective software implementations

Various attacks on IACS were reviewed in [Al20] to identify forensic traces caused by these attacks. These traces can be present in Non-Volatile Memory (Mass storage), Volatile Memory (Main memory) or Communication (Network communication). All reviewed attacks identified potential traces in the communication. This is because communication between various devices is a necessary component of all remotely executed attacks on IACS. Potential exceptions are supply chain attacks in which case a component is manipulated by an attacker before it is installed within an IACS. If the component in question can directly affect the physical process, communication with different Security Levels or Zones is not necessary.

2.3 Requirement Definition

This section discusses how the components and their communication within an IACS translate to the requirements for a modelling approach in order to increase security.

The general aim of the modelling is to describe both the current and the target state of the system. This enables system planning to be carried out more efficiently. It facilitates the creation of test cases and discovery of potential attack vectors. Also, it simplifies the risk assessment of changes and weak spots of the system.

Based on the previous considerations, we define the following requirements for modelling:

- **Technical requirements** (general modelling techniques and tools):
  - Any present physical components, logical entities and possible communication channels must be depicted.
  - The visualization must enable fast and easy comprehension of the infrastructure with its static components and dynamic behaviour.
  - Model elements must be both standardizable and customizable.
  - The resulting model files must be of reasonable size in data.
- **Subject-specific requirements** (dependent on the domain and system):
  - The PERA [Wil92] levels must be depicted.
  - The security zones and levels from the "Graded Approach" [IA11] must be depicted.

3 Modelling Aspects and Approaches

Covering security and its implications in an industrial facility requires various aspects to be explored and modelled. This chapter discusses these aspects and explores means to model them.

3.1 Modelling Aspects

To provide a complete view of the industrial facility, we define the following three aspects which must be covered by the models:

- **Static infrastructure**: These models visualize the physical components contained in the infrastructure together with their connections between each other. They include the control technology for the physical process (sensors, actuators and control devices) and network devices (e.g. switches and firewalls).

- **Physical process**: These models visualize the sequence of the physical process implemented by the control technology. They show the individual steps that happen within the process which also includes switching between physical states. This modelling aspect describes the dynamic behaviour of the system.
- **Control loop and its technology**: These models visualize the components which supervise and control the physical process. This includes sensors, actuators and their controllers as well as the communication between these devices. Hence, these models describe the IACS. This modelling aspect emphasizes the signals and information that is being exchanged between the devices as well as their consequences. It enables a more detailed reflection of attack vectors for the industrial facility. Thus, it also describes the dynamic behaviour.

Together, these three aspects allow modelling of the infrastructure and behaviour for the entire industrial facility. The distinction between physical process and control circuit is made due to the following reasons:

The physical process focuses on the sequences of physical states that are based on logical conditions and resulting actions.

The modelling of control circuits and technology then describes the implementation of these logical conditions and resulting actions with actual devices and their actual communication between each other.

This distinction allows a more fine-granular view on the infrastructure with its components and processes depending on the needs of the administrator or researcher.

However, all these models are closely interlinked with each other. The model of the control circuit includes communication with other systems. Hence, remote attacks would have to move through the modelled IACS to have an impact. In this case, the model of the physical process is necessary to understand the potential consequences of a vulnerability within the control loop. This connection allows for the identification of a safety impact due to a security problem. The modelling of the infrastructure covers supply chain attacks. Again, the connection to the control loop model is necessary to understand how such an attack could propagate within the network. The connection to the model of the physical process is again necessary to consider potential impacts on the physical process.

3.2 Modelling Approaches

This section describes two possible approaches: Manual and automated modelling.

Manual Modelling

For the process of manual modelling, we chose the tool “draw.io” (available at [Dr21]). This tool is a JavaScript based open-source software for creating diagrams. It is easy to

use, the created files are usually smaller than one megabyte and the creation of custom symbols is possible. The models are highly reusable and adaptable. Also, linking multiple models is possible. This makes the tool a practical solution for the modelling process.

The exemplary models in this paper are based on the Integrated Nuclear Evaluation System 7 (INES-7) of the Research Group of Multimedia and Security at OvGU. It is a demonstrator for the physical and control processes of power plants. In this case, the reactor is represented by a radio-controlled water heater.

One aspect of modelling is the static infrastructure. The modelling of a plant takes place in separated plant areas. That means different factory buildings are depicted in different sections of the model. Actuators and sensors control the physical process. Hence, they belong to PERA level 0 in the model. All physical and logical controllers (e.g. PLCs) are denoted as entities. Each entity gets assigned to a zone, a PERA level and a security level. Each entity has symbols assigned to it as well as communication channels drawn for it. Also, they are classified as bidirectional, unidirectional receiving or unidirectional sending. The modelling of the static infrastructure is separated into two zones: Water heater and phone (see Fig. 2). The heating control circuit consists of a sensor for the water temperature, a sensor for the water level and an actuator for the heating element. The sensors use an ellipse as the symbol, the actuator uses an adapted valve symbol.

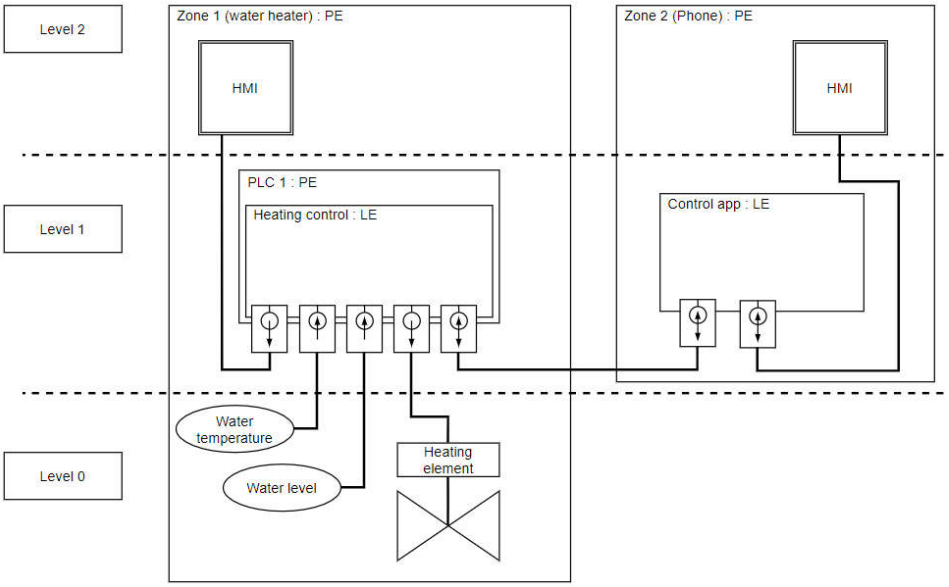

Fig. 2: Modelling of the static infrastructure of a Wi-Fi water heater

For the modelling of the physical process, a flow chart or a more complex Unified Modelling Language (UML) diagram for the process steps is sufficient. A simple flow

chart (see Fig. 3) is sufficient to show the process of heating up water.

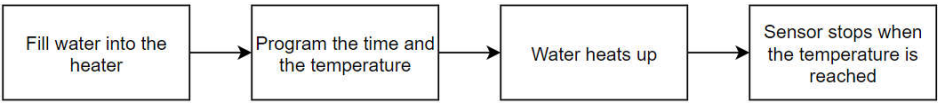

Fig. 3: Modelling of the physical process of heating water with a flow chart

In the modelling of the control loop and its technology, we differentiate between static and dynamic behaviour. In the static case, we depict each PLC individually. The model includes these elements:

- **Sensors**: Gathers information about the environment.
- **Calculation**: Transforms the sensor input into usable data representation.
- **Distribution**: The result of the calculation is distributed on the wiring.
- **Aggregation**: The data is aggregated and processed using a defined logic.
- **Logic**: Definition for the behaviour of the actuators.
- **Actuators**: Affects the environment based on the logic.

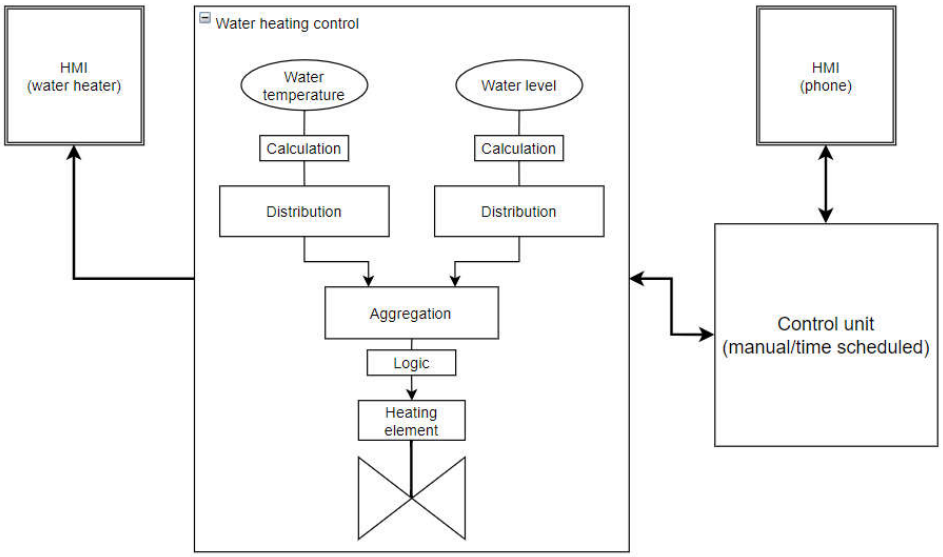

Fig. 4: Generic modelling of the control loop and its technology of a Wi-Fi water heater

The sensors and actuators are connected to the PLC with an indicator for the unidirectional communication. The physical entity of the PLC consists of the logical entity of the heating control. The PLC communicates with the second zone in a

bidirectional way. A logical entity in form of an app controls the process parameters and displays sensor values. A bidirectional channel exists for the Human Machine Interface (HMI) on the phone. Each entity is aligned according to its PERA level. The security levels could be colour-coded with different background of the entities.

Fig. 4 shows the static behaviour of the control loop and its technology. Sensors and actuators are put into a control loop with these steps: Calculation, distribution, aggregation and logic. A control unit is used as the logical entity for defining the process parameters if necessary. Additionally, the HMIs are depicted. With this simple, but meaningful example the weak spots for potential threats of a system can be assessed fast and they can be easily visualized.

Automated Modelling

Besides manual modelling, we also want to point out the possible use of automation in modelling. Creating models automatically could come with several advantages over manual modelling:

- **Less time cost for human resources**: Manual modelling of infrastructures requires an employee to invest time to create the model. Automatic modelling would only require the employee to start the process. Then, the automated modelling tool creates models based on its configuration. A final manual check might be required but would consume far less time than going through the entire process of manual modelling.
- **Gather additional information**: An automated modelling process may be able to gather new information that would have been undetected when using manual modelling with reasonable time investment. For instance, an administrator knows that he uses a specific network protocol for any communication within all applications on a device. To verify that this is the only protocol on the wire to and from this device, he looks at the network traffic for a certain amount of time with analysis tools like Wireshark [Wi21]. However, the operating system may communicate with other devices via more protocols which the administrator does not yet know about. In this case, an automated modelling tool could gather more information as it is a continuous process of gathering and modelling information.
- **Overcoming human failure factors**: Modelling a communication setup could include scanning and assessing all ports in use on a specific device. When having long lists of information (here ports), manual assessment may lead to lapse (defined in [An18]) due to human error. This can result in security issues along the modelling, development & engineering pipeline. Automation provides higher reliability and overcomes these potential failures.

4 Outlook

This paper discussed how domain specific modelling can support security in IACS by enabling the identification of vulnerabilities. To achieve this, the modelling has to fulfil domain specific requirements including the ability to model domain specific security measures. Various possible attack vectors and the complexity of IACS and the physical processes they control lead to the establishment of three types of models which are interlinked. Some basic methods to conduct such a modelling are proposed - including the establishment of specific sets of modelling symbols to describe domain specific elements.

The Research Group of Multimedia and Security at the Otto-von-Guericke-University (OvGU) of Magdeburg is currently working on two publications based on this paper. First, an automated approach on modelling will be created and evaluated. In section 3.2, the Research Group Multimedia and Security at OvGU Magdeburg has already introduced the possibility for automation. Second, the Research Group of Multimedia and Security at OvGU introduced the interlinked approach with its aspects in section 3.1 in this paper and plans to create an interlinked concept for models which combines the three aspects with regards to popular forensic models like data streams, data types and incident ontologies (see [BS11]).

Acknowledgments

The work on this paper was partly funded by the German Federal Ministry for Economic Affairs and Energy within the project SMARTEST2 (Grant No.: 1501600B).

Bibliography

[AG21] AGCS: Managing the impact of increasing interconnectivity: Trends in cyber risk. Report, Allianz Global Corporate & Specialty, 03 2021.

[Al20] Altschaffel, R.: Computer forensics in cyber-physical systems: applying existing forensic knowledge and procedures from classical IT to automation and automotive, https://opendata.uni-halle.de/handle/1981185920/35574, accessed: 17/06/2021, 2020.

[An18] Anu, V.; Hu, W.; Carver, J.; Walia, G.; Bradshaw, G.: Development of a human error taxonomy for software requirements: A systematic literature review. Information and Software Technology, 103:112–124, 2018.

[AS19] A., Saravanan; S., Bama S.: A Review on Cyber Security and the Fifth Generation Cyberattacks. Oriental Journal of Computer Science and Technology, 12:50–56, 2019

[BS11] BSI: Leitfaden „IT-Forensik“. Technical guideline, Bundesamt für Sicherheit in der Informationstechnik (BSI), Bonn (DE), 03 2011.

[Dr21] draw.io. Online: https://github.com/jgraph/drawio (Last access: 2021/06/18)

[Hi20] Hildebrandt, M.; Altschaffel, R.; Lamshoft, K.; Lange, M.; Szemkus, M.; Neubert, T.; Vielhauer, C.; Ding, Y.; Dittmann, J.: Threat Analysis of Steganographic and Covert Communication in Nuclear I&C Systems. In: In Third International Conference on Nuclear Security: Sustaining and Strengthening Efforts (ICONS 2020). 2020.

[IA11] IAEA: Computer Security at Nuclear Facilities. Technical guidance: Reference manual, International Atomic Energy Agency (IAEA), Vienna (AT), 12 2011

[Ro11] Rockwell Automation: Converged Plantwide Ethernet (CPwE) Design and Implementation Guide. Design and implementation guide, Rockwell Automation, 09 2011.

[St14] Stirland, J.; Jones, K.; Janicke, H.; Wu, T.: Developing Cyber Forensics for SCADA Industrial Control Systems. In: The International Conference on Information Security and Cyber Forensics (InfoSec). pp. 98–111, 2014.

[Wi21] Wireshark. https://www.wireshark.org/about.html, Online; Accessed: 2021-06-18.

[Wi92] Williams, T. J.: The Purdue enterprise reference architecture: a technical guide for CIM planning and implementation. 1992

Security aspects of FPGA and virtualization case studies

Asmaa Tellabi[1], Abdelbast Sabri[2], Christoph Ruland[3], Karl Waedt[4]

Abstract: Virtualization technology is a technology that has been integrated a lot lately, thanks to its interoperability and enhanced performances. An essential element in virtualization is the hypervisor that is responsible of managing virtual machines; it allows various guest operating systems (OS) to run on one hardware entity simultaneously. Many researchers recognized the challenge of allowing multiple critical applications to share one hardware platform without interfering, and ensuring security of transactions a major safety and security challenge in virtualization. Usually, security by design is also comprises the protection of intellectual property (IP), possible safety related damages and extend financial losses. By the extension of the usage of programmable logic outside commercial markets to avionics and military applications, security by design integrates extra aspects to ensure safety and cybersecurity. Solutions for protecting application data during transmission and storage exist, but solutions for protecting Field Programmable Gate Array (FPGA) configuration data are not well-known. In the context of Industry 4.0, the FPGA hardware solutions provide the needed level of flexibility and performance. The flexibility, portability and even, to a wide extent, technology independence are due to the software based definition of the hardware via Hardware Description Languages (HDL).

In this paper, threats and vulnerabilities surrounding FPGAs will be addressed. An example of a type 1 hypervisor called XtratuM will be ported on top of a Xilinx Soc 7000 zc706 will be integrated, as well as a prototype of embedded system using PetaLinux will be provided.

Keywords: Virtualization, hypervisor XtratuM, embedded system using PetaLinux, FPGA Xilinx Soc 7000 zc706.

1 Introduction

Currently, the global marketplace has presented multiple novel opportunities but also emerging threats. Corporations and governments must cope with these threats that vary from counterfeiting till espionage. Consequences of such threats can go far beyond financial losses as they can also harm personal safety [JP09]. Enterprises deploy various layers of protection to ensure security; they contain firewalls, and authentication/encryption, security protocols and intrusion detection/intrusion prevention systems [IC19]. Cybersecurity has been a critical subject for many companies for over 25

[1] University of Siegen, Faculty of Science and Engineering, Chair for Data Communication Systems, Hölderlinstraße 3, Siegen, 57068, asmaa.tellabi@student.uni-siegen.de
[2] University Friedrich-Alexander-University, Martensstraße 5a, 91058 , Erlangen, abdelbast.sabri@fau.de
3 University of Siegen, Faculty of Science and Engineering, Chair for Data Communication Systems, Hölderlinstraße 3, Siegen, 57068, christoph.ruland@student.uni-siegen.de
[4] Framatome GmbH, Henri-Dunant-Str. 50, 91058, Erlangen, karl.waedt@framatome.com

years; although it is only recently that it caught engineers' attention working on embedded systems [IC19].

Easily reconfigurable architectures such as FPGAs are offering multiple remarkable features that can be integrated in embedded systems in case security is a vital matter [Ge12]. FPGAs from 2007´s came out with multiple functional features than the ones created in the mid-1980s. These new FPGAs integrate embedded processors, Giga-bit serial transceivers, clock managers, analogue to digital converters, digital signal processing blocks, Ethernet controllers, large memory capacity. Such evolution in the capacity and application levels of FPGAs has two main security consequences. To start with, current FPGA designs display a major development investment that must be protected. In addition, FPGAs are progressively more being deployed in applications that necessitate security features that are does not existing yet, or that must be sufficiently inspected. Together, they have shed the light on security attributes related to FPGAs in the military, automotive, consumer industries, and also within the research community, each with its specific requirements and security perceptions [ST12].

2 Assessment of Threats and Vulnerabilities surrounding FPGA

FPGAs are based on programmability and a set of identical logic blocks in order to produce a flexible computing entity, which is can lower design costs, decrease system´s complexity, and reduce the time to market, by means of parallelism and hardware acceleration to reach better performances. FPGAs rising popularity has pushed integrators to integrate security controls in the design´s level, but the way resources in embedded systems are allocated forms challenges in offering a secure entity [EE19]. FPGAs are programmed using a bitstream, which is a binary data that is passed to the FPGA by means of precise I/O ports on the device. They specify the way internal resources are going to be used for executing logic operations [ST12].

2.1 The Threats

There is a wide range of threats surrounding the design and each one has its particular implications. Some threats aim to gain financial benefits of a company, while other might aim to harm individuals or even national security [ST12]. Industries are slowly gravitating towards FPGAs thanks to their programmability´s benefits and proof against obsolescence. FPGAs are less susceptible to reverse engineering attacks but to another different kind of threats [Zh18].

- **Reverse Engineering**

Using an already existing product in the market, attackers analyze the design by observing the layout, the used devices, downloading the firmware, and examining transactions

between devices. By exploiting this information, attackers might rebuild the design, so they can create new products or help their future products development [ST12].

- **Cloning**

For this type of threats, attackers do not try to fully comprehend and analyze the design. Their goal is to produce replicas of a current product, and realize a better profit by selling it since they have not went through the same the time and expense of product development and marketing processes [SD08]. As attackers are less likely to use a better quality modules and quality assurance components, the developed replicas can harm the company´s reputation and also its finances [ST12]. Serious impacts happen when cloning is used to create replicas products dedicated to high availability applications, e.g., in aviation which can negatively impact flight´s safety [Zh18].

- **Overbuilding**

One of the easiest and simplest types of design´s theft is overbuilding. With the growing trend of outsourcing, usually original equipment manufacturers (OEM) count on off-shore suppliers to create its products [SD08]. As a result, a dishonest subcontractor might create some additional components more than those ordered by the OEM. Even though this can be seen as a type of counterfeiting, these extra components that are produced are similar to the originals, which make the detection challenging [Zh18].

- **Tampering**

This refers to an outside attacker trying to gain illegal access to an electronic device. Tampering might be either a part of a reverse engineering attack, or it can aim to realize malicious or criminal acts [ST12]. For example, an attacker can attempt to mine operating data or firmware, or can attempt to alter the firmware in the system so they can put the system in jeopardy [SD08].

- **JTAG Threats**

A different kind of threats exploits the JTAG port. Even if the port is used for board connectivity and to test system´s functionalities, these ports can be exploited to conclude the FPGAs configuration [ST12]. JTAG is always set up in most FPGAs, it can be used to apply the design methodically so it makes reverse engineer attacks easier, thus stealing the design´s information [Zh18]. On the other hand, similar to timing analysis on an FPGA, this type of attacks consumes a lot of time and requires specific skills but it does not require dedicated equipment [SD08].

2.2 Weaknesses

- **Complacency**

This occurs in the design teams and companies´ side. Some companies forget about integrating security in the design because it requires more time or to the certainty that legal

protection is enough [Ge12]. Therefore, integration of security in the design is not advised, which make companies consider only few steps to protect its valuable intellectual property [ST12]. This type of weakness can be mitigated by proper training, education and the acknowledgment that even if a legal protection exists it does not ensure entire protection [Zh18].

- **Inadequate Security Controls**

This represents another category of vulnerability at the FPGA level [EE19]. For instance, a company can integrate an anti-tamper detection device in a system including an FPGA to warn end users of tampering attacks, but in case these security controls do not include encryption of FPGA bitstreams the system will still be exposed to reverse engineering attacks [Zh18]. Furthermore, security controls must not be included just at the device or FPGA level, but also at the board and system levels, taking into consideration possible threats at each level. Obviously, designs need to be carefully reviewed to guarantee that all security characteristics are examined [ST12].

- **Back Doors**

Some extra configurations applied in a design to help with the debug might create security holes that can be exploited by attackers. From analogous to software systems, some back doors are left to facilitate the access to system administrators, but the hardware design can have some security holes as well [Ge12][ST12]. For instance, in case device debugs modes/cores they can be exploited in order to go beyond security controls and anti-tampering measures. Although these back doors can be helpful in the period of the design, they have to be cut out from the design before final assembly.

- **Design Flaws**

These flaws can create exploitable security holes, e.g., untested cases that are present in the design can make it vulnerable [XW19]. Consequently, careful attention must be given to testing before delivering the final product [ST12].

- **Device Flaws**

Same as design flays, a device can have manufacturing flaws that make it susceptible to an attack. Choosing vendors with in-depth testing structures and unconventional quality assurance techniques can significantly reduce this risk [XA19].

- **Single-Event Upsets**

Single-event upsets (SEU) arise when device memory structures have their states modified because of the impact of high-energy neutrons. An SEU is able of modifying the functionality of a device and compromising security controls included inside the device [ST12]. Applying SEU mitigation methods not only can neutralize this risk but it can enhance system reliability.

Virtualization use case study

Xtratum is a bare-metal or type 1 hypervisor, it was conceived especially to adhere to temporal and spatial requirements for safety critical applications [FE18]. Xtratum offers virtualization extensions to partitions; it runs in supervisor processor mode and virtualizes the CPU, memory, interrupts, and some defined peripherals [Al15]. The first version was intended for x86 Architectures (version 2.0), then it has been completely transformed to be compatible with SPARC v8 architecture, to be exact for LEON2 processor. Present versions mainly include the necessary functionalities required to create safety Critical systems based on ARINC 653, AUTOSTAR and other standards [As18].

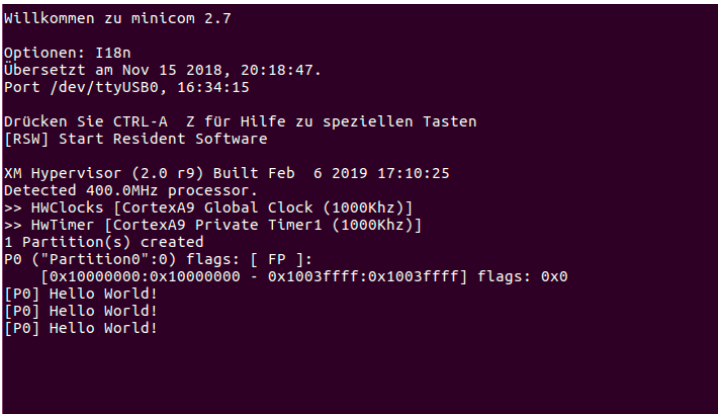

Fig. 1: Execution Results of the ´Hello World´ Partition.

```
<?xml version="1.0"?>
<SystemDescription name="hello_world" version="1.0.0" xmlns="http://www.xtratum.org/xm-arm-2.x">
  <HwDescription>
    <MemoryLayout>
      <Region size="1MB" start="0x0" type="rom"/>
      <Region size="1023MB" start="0x00100000" type="sdram"/>
    </MemoryLayout>
    <ProcessorTable>
      <Processor frequency="400Mhz" id="0">
        <CyclicPlanTable>
          <Plan id="0" majorFrame="200ms">
            <Slot start="0ms" id="0" partitionId="0" duration="200ms"/>
          </Plan>
        </CyclicPlanTable>
      </Processor>
    </ProcessorTable>
    <Devices>
      <Uart name="Uart" id="1" baudRate="115200"/>
    </Devices>
  </HwDescription>
  <XMHypervisor console="Uart">
    <PhysicalMemoryArea size="1MB"/>
  </XMHypervisor>
  <PartitionTable>
    <Partition name="Partition0" id="0" console="Uart" flags="boot fp">
      <PhysicalMemoryAreas>
        <Area size="256KB" start="0x10000000"/>
      </PhysicalMemoryAreas>
    </Partition>
  </PartitionTable>
</SystemDescription>
```

Fig. 2: XML file of the ´Hello World´ Partition.

In this paper, the version used of Xtratum is the one compatible with Arm processors [ARM18]. The ZC 706 board includes 2 Cortex A9 processors, in this example only one processor will be used. Xtratum has a default configuration made for this board specifically [As18a]. In this implementation, a hello world application is implemented. To be able to run application on Xtratum, an XML file has to be first configured, this file include all physical resources, and a scheduling plan for partitions [FE18]. To compile applications on the board, the command make is used. Figure 1 shows the results of the running application on the board.

2.3 PetaLinux

PetaLinux Tools offer the necessary tools used to customize, build and deploy Embedded Linux solutions on Xilinx processing systems. Designed to speed up design productivity, these tools are used together with the Xilinx hardware design tools to facilitate the development of Linux systems for Zynq-7000, MicroBlaze and PowerPC [XW19]. PetaLinux can be used by developers to configure, build and deploy the required open source and systems software to Xilinx, including [XA19]:

First Stage Boot Loader (FSBL) : is responsible of loading partitions (software programs, the bitstream) in the image, that is stored in Non-Volatile Memory (NVM) , to the destination partitions. Usually, the destination can be Double Data Rate (DDR), On-Chip Memory (OCM), or Advanced eXtensible Interface battery-backed RAM (AXI BBRAM). The destination of the bitstream is the PL configuration memory

U-BOOT: which is an open source software that is executed on Zynq devices. It is generally used to load Linux. Other U-Boot functions contain reading DDR memory, erasing, reading and writing into NVM.

Linux kernel: is the core of the Linux OS, it is responsible of controlling operations in the system. Kernel.org together with Xilinx additions (BSP and drivers) provides the Linux kernel for Xilinx Zynq platforms. Linux kernel is the main component of the OS that connects the hardware to applications.

The device tree: is basically a data structure in byte code that contains helpful information used by the kernel when booting up. The boot loader copies data into a fixed address in the RAM before jumping to the kernel's entry point.

Root filesystem (Rootfs): comprises various specific configuration files related to systems. Some examples contain a kernel that is created for a system, a specific hostname, etc. Their contents have to be suitable to boot, restore, recover, and/or repair the system. To boot a system, a certain amount of software and data must be included on the root partition to mount other filesystems. This consists of utilities, configuration, boot loader information, and other necessary start-up data.

First, the FSBL is executed followed by the execution of U-Boot, which loads the device tree and Linux kernel into the memory and then mounting rootfs are executed. After the hardware bit stream and software images have been built, the new PetaLinux platform with the Zynq kernel can be booted via an SD card. Figure 1 shows the design flow on Vivado and PetaLinux.

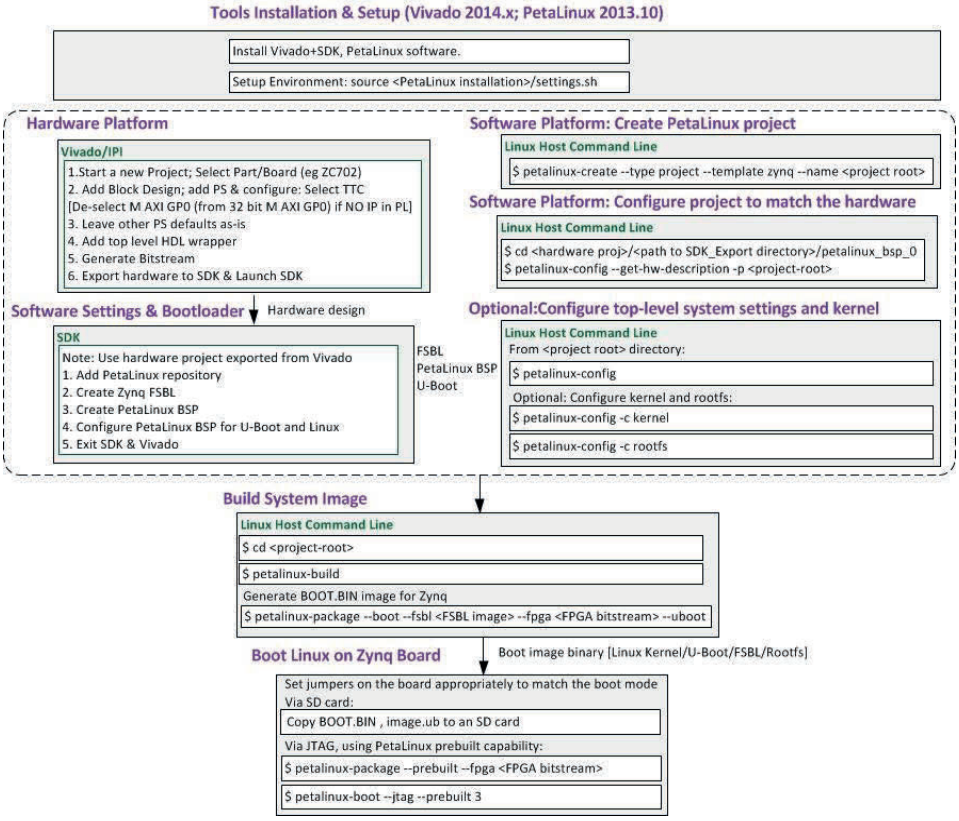

Fig. 3: Design flow using Vivado (v2014.x) and PetaLinux (v2013.10)[Ja19].

The Xilinx Vivado Software Development Kit (SDK), PetaLinux tools and Board Support Package (BSP) is a set of customized drivers to the selected hardware; they must be installed before any design development. PetaLinux and Vivado SDK must be of the same version, e.g. if Vivado and SDK 2018.2 are installed, the PetaLinux version must be 2018.2. In this example booting Linux on Zynq can be done using JTAG or SD card [XA19][XW19].

The petalinux-create command creates partitions that are part of a PetaLinux project, a partition can be either a new project, a component or an application in the same project [PC19]. Components or applications can then be either integrated or excluded from the final system using the petalinux-config -c rootfs command. Dropbear is a similar tool to OpenSSH . To use openssh-server, dropbear must be set up in the PetaLinux Root File System before applying the petalinux-build using the following command: petalinux-config -c rootfs. This will configure the project to use SSH between the board and other external systems [XW19].

In this paper, 2 applications were developed and tested in PetaLinux.

The first one is a Hello World application that was created using Petalinux 2018.02. PetaLinux provides a tool to create user application templates for either C or C++ [PC19]. These templates include applications´ source code and Makefiles to facilitate the configuration and compilation of applications in the target, and then installing them into the root file system. The new application can be found in the <project-root>/components/apps/myapp directory.

A terminal emulator program should be configured to connect to the USB serial port at 115,200 baud. The emulator minicom is used in this example to connect to the serial port. The second one is related to OpenSSH. With SSH into the board can files be edited and compiled on the desktop, and then executables copied over Ethernet. When the ifconfig is executed in the emulator, first the inet addr shows which represents the ip-address of the board, in this case 192.168.0.10. SSH can be activated by executing the following command: ssh root@192.168.1.69 in a new terminal.

3 Conclusion

Virtualization technologies introduce new threats, risks, challenges, new assets and components to the digital environment. Consequently, novel security controls and measures are necessary to offer a trustworthy system for such environments. Multiple techniques and applications for different types of virtualization exist; but they all have a same objective which is to erase the limitations present in the physical hardware by building secure partitions in the hardware, which make it operates like various virtual environments. This paper listed most of the common vulnerabilities and weaknesses found in FPGAs. An example of a Type 1 hypervisor running on top of the board is presented as

well. An illustration of embedded systems procreation using PetaLinux on a Xilinx ZC 706 board is shown. In the future, more hypervisors such as Xen are going to be tested on the board and other boards with different architectures like x86.

Bibliography

[JP09] Juwayriyah, H; Paul, Q.: Protecting the FPGA Design from Common Threats; 2009.

[ST12] Steven, M: Solving Today's Design Security Concerns; 2012.

[Ge12] Gedare, B.; Eugen, L.: Bhagirath, N.: Rahul, S.: Hardware and Security: Vulnerabilities and Solutions. In (Elsevier Inc.): Handbook on Securing Cyber-Physical Critical Infrastructure, New York 1999. Noah & Sons, San Francisco, pp. 305-331, 2012.

[IC19] Icon Labs: Security Requirements for Embedded Devices – What is Really Needed?, www.iconlabs.com, accessed: 21/02/2019.

[EE19] EETimes: New Advances in FPGA Security, https://www.eetimes.com/document.asp?doc_id=1327477#, accessed: 04/05/2019.

[Zh18] Zhiming, Z.; Laurent, N.: Charles, A.K.: Qiaoyan, Y.: Thwarting Security Threats From Malicious FPGA Tools With Novel FPGA-Oriented Moving Target Defense, 2018.

[SD08] Saar, D.: Solving Today's Design Security Concerns; 2008.

[XW19] Xilinx Wiki, https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842250/PetaLinux, accessed: 30/04/2019.

[XA19] Lester Sanders: XAPP1175, Secure Boot of Zynq-7000 All ProgrammableSoC, 2015, https: //www .xilinx. com/ support /documentation/ application_notes /xapp1175_zynq_secure_boot.pdf , accessed: 30/04/2019.

[FE18] Fentiss: Fent Innovative Software Solutions, www.fentiss.com/xtratum, accessed: 20/08/2018.

[Al15] A. Alonso; et.al ,: Safety Concept for a Mixed Criticality On-Board Software System .IFAC-PapersOnLine, Volume 48, Issue 10, Pages 240-245, ISSN 2405-8963, 2015.

[As18] Asmaa, T.; Ines, B.Z: Christoph, R.: Karl, W.: Virtualization on Secure Platforms for Industrial Applications Current use cases and future perspectives, 2018.

[ARM18] ARM: ARM Security Technology - Building a Secure System using TrustZone Technology. Tech. Rep., 2009.

[As18a] Asmaa, T.; Ludger, P.: Christoph, R.: Karl, W.: Security Aspects of Hardware Virtualization Technologies for Industrial Automation and Control Systems, 2018.

[Ja19] Janani Subraveti: Implementation of UDP communication on a ZYNQ platform for processing clustered data based on multiple Gigabit Ethernet ports for phenoPET, 2017, https://juser.fz-juelich.de/record/844070/files/J%C3%BCl_4406.pdf, accessed: 30/04/2019.

[PC19] PetaLinux Command Line Reference UG1157, https: //www .xilinx.co m/support/documentation/sw_manuals/xilinx2018_3/ug1157-petalinux-tools-command-line-guide.pdf, accessed: 30/04/2019.