IoT4CPS – Trustworthy IoT for CPS

FFG - ICT of the Future
Project No. 863129

Deliverable D3.3
Guidelines and recommendations for resilient system architecture pattern and concepts and HW-based solutions for safe & secure IoT

The IoT4CPS Consortium:
AIT – Austrian Institute of Technology GmbH
AVL – AVL List GmbH
DUK – Donau-Universität Krems
IFAT – Infineon Technologies Austria AG
JKU – JK Universität Linz / Institute for Pervasive Computing
JR – Joanneum Research Forschungsgesellschaft mbH
NOKIA – Nokia Solutions and Networks Österreich GmbH
NXP – NXP Semiconductors Austria GmbH
SBA – SBA Research GmbH
SRFG – Salzburg Research Forschungsgesellschaft
SCCH – Software Competence Center Hagenberg GmbH
SAGÖ – Siemens AG Österreich
TTTech – TTTech Computertechnik AG
IAIK – TU Graz / Institute for Applied Information Processing and Communications
ITI – TU Graz / Institute for Technical Informatics
TUW – TU Wien / Institute of Computer Engineering
XNET – X-Net Services GmbH

For more information on this document or the IoT4CPS project, please contact:
Mario Drobics, AIT Austrian Institute of Technology, mario.drobics@ait.ac.at

Document Control

Title: Architecture for safe and secure automated driving platform demonstrator
Type: **confidential**
Editor(s): **Edin Arnautovic**
E-mail: **edin.arnautovic@tttech.com**
Author(s): Edin Arnautovic, Ralph Ankele, Leo Happ Botler, Denise Ratasich, Ezio Bartocci, Jakšić Stefan
Doc ID: **D3.3**

Amendment History

Version	Date	Author	Description/Comments
V0.1	30.04.2019	Edin Arnautovic	Initial version prepared
V0.2	30.04.2019	Ralph Ankele	Initial Recommendations and Requirements to interlink WP3 models and WP4 security assurance tools.
V0.3	08.05.2019	Leo Happ Botler	Initial trusted localization
V0.4	17.05.2019	Denise Ratasich, Ezio Bartocci	Architectural Aspects of Self-Healing by Structural Adaptation
V0.5	17.05.2019	Ralph Ankele	Final Recommendations and Requirements to interlink WP3 models and WP4 security assurance tools.
V0.6	27.05.2019	Jakšić Stefan, Abdelkader Shaaban	Security analysis of components in a model of device connect use case
V0.7	29.05.2019	Leo Happ Botler	Trusted Localization
V0.8	21.06.2019	Edin Arnautovic	Pre-final for review.
V0.9	27.06.2019	Stefan Jakšić, Violeta Damjanović-Behrendt	Review editing
V1.0	29.96.2019	Edin Arnautovic	Final version

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

Federal Ministry
Republic of Austria
Transport, Innovation
and Technology

FFG
Forschung wirkt.

Content

Executive Summary

This deliverable deals with resilient IoT systems by offering guidelines and patterns for developing and assessing safety and security features more efficiently. On the safety side, it takes an approach of separating safety-relevant system features from the concrete functions (applications) in a form of a safety platform. The platform should offer the abstraction of underlying hardware and safety measures and such give the application developers an effective way to develop application without investing a lot of effort on the safety considerations. On the security side, this deliverable analyses security vulnerabilities and offers concepts and patterns for security assurance, e.g. through threat modelling, or security testing.

1. Introduction

This deliverable deals with concepts and patters for safe and secure Internet of Things (IoT). These concepts and patterns are *technological building blocks* which address the business needs defined in IoT4CPS deliverable D2.2 (WP2). For example, the Use Cases of *Level 3 and Level 4 Automated Driving* require a mixed-criticality safety platform architecture. This requires safety-relevant system features to be separated from the concrete functions (applications) in a form of a *safety platform*. The safety platform should offer the abstraction of underlying hardware and safety measures and give the application developers an effective way to develop application without investing a lot of effort on the safety considerations. Furthermore, the safety platform guarantees real-time execution and communication, freedom of interference in embedded systems, etc. A pattern for increased reliability by replacing a failed component during runtime is known as *Self-Healing by Structural Adaptation.* From the security perspective, this report analyses security vulnerabilities and offers concepts and patterns for security assurance, e.g. through threat modelling, or security testing.
The business need of *security verification along the full life cycle* requires *Security Assurance through Threat Modelling, Security Analysis and Penetration Testing.* When IoT systems are analysed for security vulnerabilities, the more information that is available to an attacker, the more security vulnerabilities can be identified. Task 3.2 of WP3 of the IoT4CPS project aims to identify system architecture patterns and concepts for safe and secure IoT solutions by defining a model-based approach of the IoT system. Task 4.1 of WP4 of the IoT4CPS project aims to verify and analyse the security of those models. In this task we want to identify the security related properties in the WP3 models and give a recommendation on what properties the WP3 models must include to provide all necessary input parameters of the WP4 security assurance tools (for threat modelling/penetration tests).
The overall goal is to interlink the WP3 models and the WP4 security assurance tools. In the following, we first classify the typical IoT/ Industrial Internet of Things (IIoT) components. Secondly, we provide a brief overview of the security assurance tools that are used in WP4. Next, we provide a list of desired properties for the prior identified IoT/IIoT components that will later provide the necessary information to efficiently automate the security analysis and verification of the IoT/IIoT system under test. Finally, we provide an example according to the AVL industrial “Device.CONNECT™” use case.
With the digital interconnection of smart devices, sensors, embedded systems, home automation systems and others, the IoT was born. In parallel, companies optimise their manufacturing processes by interconnecting sensors, instruments and other devices networked together within large industrial applications, that is often referred to as IIoT or Industry 4.0. While the general concept of IoT and IIoT are similar, often the complexity of the so-called smart home vs. smart factory is different. The following gives a classification of the different IoT and IIoT devices and components.

1.1 Smart Devices and Sensors

IoT systems consist of end-user devices and sensors used to obtain information. IIoT systems often contain many Cyber-Physical Systems (CPS), which integrate physical systems to software and communication systems. These devices include, for example, temperature sensors, thermostats, pressure sensors, humidity and moisture level sensors, light intensity detectors, proximity detection, RFID and others.

1.2 Networks and Protocols

The IoT requires huge scalability in the network space to handle the vast number of interconnected devices. With billions of devices being added, communication networks and protocols must be adapted to handle the data flow. While general networks like Ethernet (i.e. the address space of IPv4 allows 2^{32} = 4.3 billion devices, while estimates show that there will be around 30 billion IoT devices in 2020 [ST19]) and WIFI are not optimized for IoT use cases [SYDZ16], different networks and protocols emerged like Bluetooth LE[B19], ZigBee[Z12], MQTT[BG14], Z-wave[S19], LoRAWAN[L17], IPv6[DH98], and others.

1.3 Gateways

In IoT networks many different networks and protocols have to interact with each other. IoT gateways can be configured to manage the bidirectional data traffic to ensure the interconnectivity of devices and sensors as well as the compatibility of network protocols.

1.4 Cloud / Servers

The IoT generates massive amounts of data from devices, sensors, applications and users that have to be managed efficiently. IIoT networks often require real-time analysis of sensor data and accurate analytics at different levels (edge, fog, cloud). The cloud is often further used to provide data storage, analytics, infrastructure and services to billions of interconnected devices.

1.5 Analysis / Actuators

Based on the sensor data, several actuators in IoT networks can perform services and further cloud-based services can produce real time insights. Big enterprises collect massive amounts of data that needs to be carefully analysed according to the business use cases. In IIoT networks several sensors provide information that needs to be combined and analysed.

1.6 User interface

User interfaces are the accessible parts of the IoT that connect the user with the devices. In IIoT networks the user interfaces are usually replaced by well-defined APIs that offer an interface for other smart services and backend systems.

1.7 Smart Services and Backend Systems

While IoT systems often outsource the whole infrastructure, computation and storage into the cloud, IIoT systems often process information in backend systems within the network of a company. Therefore, the interplay between cloud-based services and smart services in the backend of a company has to be enabled.

1.8 Big Data Analytics / Artificial Intelligence / Machine Learning

The sensors and devices connected to IoT and IIoT networks produce vast amounts of data that need to be efficiently processed and analysed. Artificial intelligence (AI) is a field of computer science that describes machines that mimic cognitive functions of humans such as learning and problem solving. Machine learning (ML) is a core part of AI that allows software to predict outcomes and to find patterns without being explicitly programmed. Both AI and ML techniques can be used in IoT networks to efficiently analyse data.

2. Mixed Criticality Platform Architecture – Patterns and Concepts

Modern IoT devices and systems (especially in industrial or automotive context) must guarantee safety and security, as well include high-performance computing capabilities. These requirements could be conflicting. Especially, today's systems on chip are highly specialized and offer either high computing performance (e.g., with multi-core, multi CPUs on a single chip, GPUs, etc) or safety features (e.g., Lockstep CPU cores with clock delay, safety management unit, clock and voltage monitors, etc.). So, to offer both high-performance and safety features to applications, software platform solutions are necessary. An essential property of such a platform is called the *mixed criticality.* Mixed-criticality systems can execute applications with different criticality levels. The platform provides guarantees that the applications characterized with different criticality levels do not influence each other: measures to ensure *freedom-of-interference* have to be engineered in the platform. The applications must be separated booth in the terms of space (memory) and time.
The platform concepts developed in IoT4CPS address this through different patterns and modules. For example, *self-monitoring* manages different stages of runtime testing during the lifecycle of a device (an ECU in the automotive case). Typical stages are *init, running, and shutdown.* The specifically defined tests will execute in these stages to show the fulfilling of the assumptions of safety features defined during the safety analysis. Another safety-related pattern is the *tasks monitoring.* Task monitoring verifies at runtime that the behaviour of each task (or computing process) is conform to the behaviour defined in the modelling phase. Task monitoring observes the freedom of interference in time by measuring and detecting time violation (e.g., a task execution takes too long), and in memory by monitoring the memory area accessed by tasks. *Host supervision* monitors the Systems-on-Chip (SoCs) by using question/answer protocol. The answers are calculated by the "slave" SoCs and are checked by the (safety) master SoC. To enable real-time properties in a distributed computer system, all system components (e.g., CPUs) must have the same notion of time, e.g. so-called *shared time base.* Thus, the system is capable to provide guaranteed latency in both execution as well as communication across the system. For that purpose, the system components have to be *synchronized*. The approach for synchronization is based on generalized Precision Time Protocol (IEEE 802.1AS) where a *grandmaster* provides a central clock. The global time is synchronized on each host to the internal Ethernet network time and is increased monotonically. Applications use the global time to timestamp messages and calculate time intervals.

3. Architectural Aspects of Self-Healing by Structural Adaptation

Self-Healing by Structural Adaptation (SHSA) enables self-healing in a CPS given some redundancy in the messages communicated over the network exists. An overview to the architectural requirements is given in Table 1.

Table 1: Requirements to deploy SHSA on a platform.

No.	SHSA Requirement	Rationale
1	Dynamic Composability	Add substitutes, remove/replace faulty components
1a	Reconfigurable Information Flow	Reconfigurable sender/receiver of messages
1b	Common Communication	One interface to access information
1c	Freedom of Interference	Adding a substitute shall not alter the system

1d	Fault Containment	Avoid fault propagation
2	Information Access	For learning, monitoring and substitution.

In [RHISG2017] the rationales of these requirements have been discussed. However, the implementation of these needs might generate additional vulnerabilities. Some of the requirements simplify or enable attacks. Some others complicate or mitigate attacks. Here we present the outcome of a threat analysis of SHSA that is discussed in [STRIDE][KMLS2017]. Appropriate countermeasures have to be installed to ensure the desired and necessary security properties listed below.

1. Once the system is capable of starting new services an attacker might spoof this process. For instance, an attacker could start a service to subscribe to data to work out a strategic attack. The attacker could also trigger the adaptation degrading the system's performance by, e.g., spoofing the monitor trigger message. Dynamic composability is a *process* and as such vulnerable to all kind of threats [KMLS2017]. All desired security properties are therefore relevant: confidentiality, integrity, availability (CIA), authenticity, authorization (access control), non-repudiation.
    a) In [RHISG2017] we proposed to use a data-centric communication to enable adaptive information flow. For instance, in the current middleware that is based on Robot Operating System (ROS), the subscriber to specific data does not care about the sender of the messages, and is interested only in the data. An attacker could inject malicious messages to such a data channel without the receiver noticing it (apart from man-in-the-middle (MITM) or spoofing/masquerade attacks). Appropriate authorization has to be installed to restrict publishing and subscribing of specific data and to ensure authenticity and non-repudiation.
    b) Common communication is not a strict requirement but makes the reconfiguration of the information flow easier or less complicated. Once the attacker has access to the common network it should not mean it has access to all information. This architectural requirement makes it easier to attack the data flow. Relevant threats are tampering, information disclosure and denial-of-service. Desired security properties are therefore integrity, confidentiality, availability and privacy.
    c) Freedom of Interference is needed to ensure the consistency of the system before, during and after self-adaptation. For instance, when adding a component the timeliness of the system shall not be violated. When this requirement is met, even when the attacker gains permission to start processes, it does not influence the system's functionality.
    d) Components usually trust their inputs, i.e. have no monitor for inputs installed. Hence, failures may propagate through the communication subsystem, reach the physical system via actuators and cascade back to the cyber-system via sensors. However, when this requirement is met, even when an attacker breaks one component, it does not break the whole system.

Information access shall be limited only to trusted services. Access to the information distributed in the network may enable strategic attacks, specifically. A relevant threat here is information disclosure, and the desired security properties are confidentiality and privacy.

4. Security analysis of Device Connect use case using MORETO

In this deliverable we report on the security analysis which was conducted on a model of a Device Connect, a reference device for a use case in the IoT4CPS project. Our aim is to demonstrate security requirements tool called Model-Based Security Requirement Management Tool (MoReTo) and present the outputs of the requirements analysis. The MORETO tool can be used to verify security of a specific component, which then can be considered as a building block in our resilient system architecture. MoReTo contains an encoded security requirements database built upon extensive expert knowledge and can be applied for defining the security benchmark of IoT components.

MoReTo copes with the ambiguity of understanding the system security requirements by managing a massive number of security countermeasures and determines the security requirements needed to achieve a high degree of security assurance. That can be applied in various industrial perspectives such as Industrial Automation Control System (IACS), Cyber-Physical System (CPS), IoT, Automotive domains, and others. It generates a list of security requirements of the user's elements in each diagram, which can help the user to build-up a secure infrastructure - the security requirements created concerning different properties and security flaws in provided elements.

Figure 1 shows the process of modeling the Device Connect use case and its requirements in MoReTo.

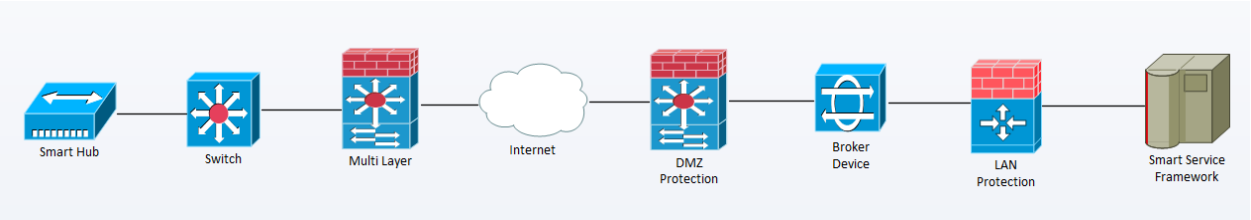

Figure 1 – AVL Device Connect use case modelled in MoReTo

MoReTo generates a list of security requirements based on expert knowledge database and on the description of the IEC 62443-4-2 standard [26]. This standard defines Components Requirements (CR) for four types of components. The CRs for each type of components will be considered as follows [1]:

- Software Application Requirements (SAR).
- Embedded Device Requirements (EDR).
- Host Device Requirements (HDR).
- Network Device Requirements (NDR).

The following sections show the output of MoReTo, which are here presented as the security requirements for each device involved in this use case.

Smart Hub device

MoReTo stipulates eight CRs of security requirements which are necessary to be considered for the Smart-hub device, as depicted in Figure 2. It can be seen that all the requirements are of type ‘‘component requirements’’ and all of them are independent.

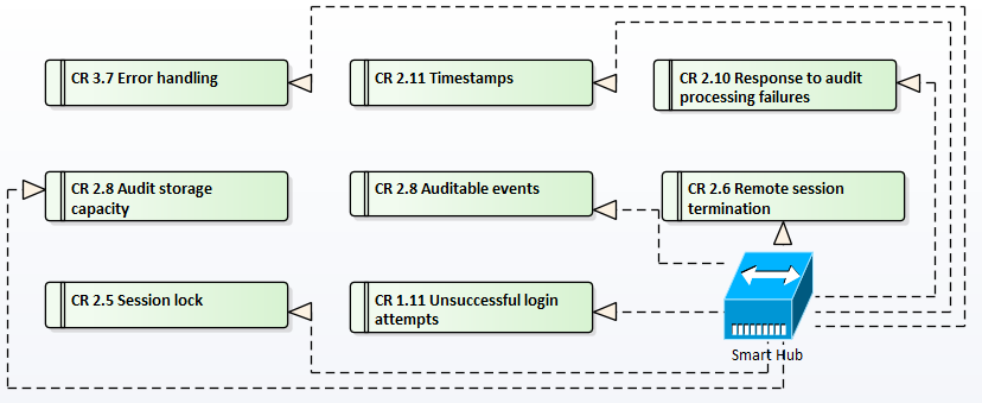

Figure 2 Security requirements of the Smart Hub in MoReTo

Switch component

Figure 3 shows the chosen security requirements for the Switch device by MoReTo. It defined two EDRs, one HDR, one NDR, and three CRs which are required to ensure the security assurance of the Switch device. For example, for the switch component it is crucial to enable support for secure updates. It can be noted that this requirement relates to different types of components: host device, embedded device and network device.

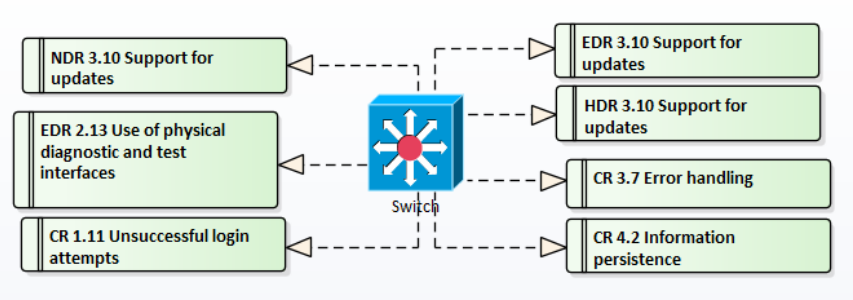

Figure 3 Switch device with the selected security requirements

Multilayer Firewall & DeMilitarized Zone (DMZ) Protection Devices

The Multilayer firewall and DMZ protection devices play an essential role in this use case, which monitor and control outgoing and incoming network traffic based on predetermined security conditions. MoReTo defines 19 different security requirements for these two devices as depicted in Figure 4.

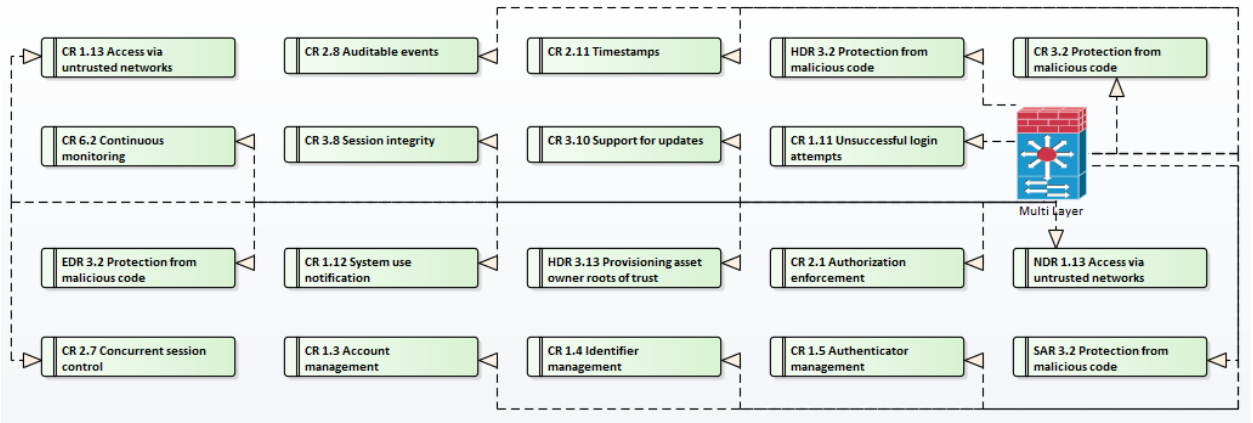

Figure 4 Security requirements of Multilayer device

Broker Device

Likewise, MoReTo provides a new list of security requirements for the Broker device.

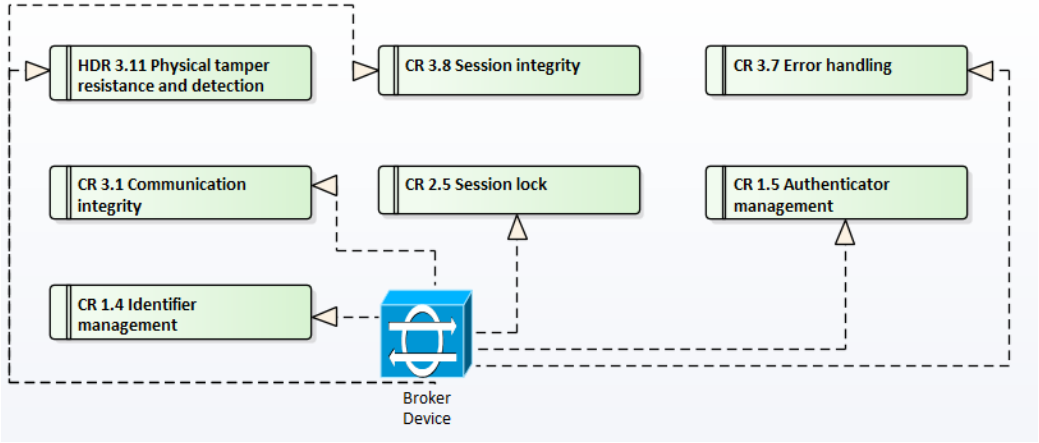

Figure 5 Broker device and list of selected security requirements

LAN Protection

The LAN Protection Device checks the network availability and the security of resources in a local network. MoReTo selects different types of security requirements for the LAN Protection Device, in order to certify the demanded security level. Figure 6 illustrates 14 security requirements of the LAN Protection device.

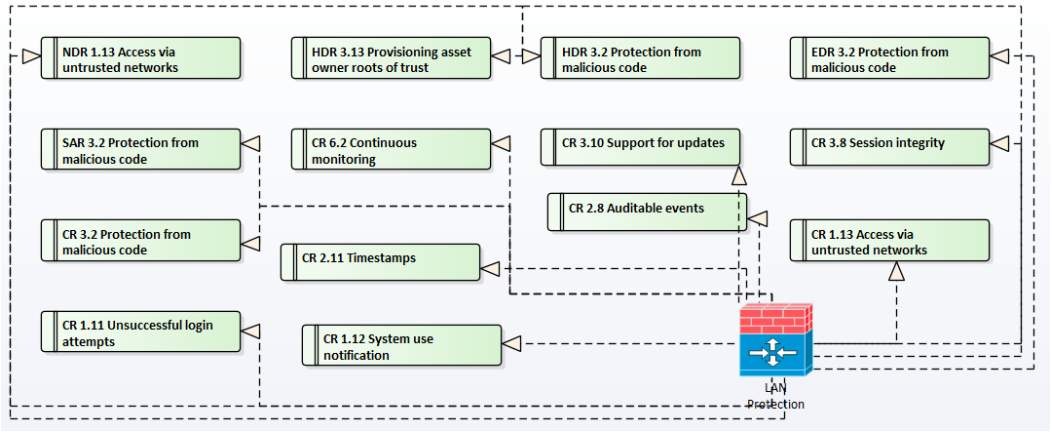

Figure 6 LAN Protection Device

Evaluation results

To summarize, MoReTo has been applied to the Device Connect use case for security requirements management process, to satisfy the evaluation of demanded security protection level. MoReTo designates the chosen security requirements according to the IEC 62443-4-2 security types. Figure 7 shows, the rate of each security requirements as maintained by security groups. For the simplicity and readability of this deliverable, the results of security analysis for the Smart Service Framework are omitted. The most of the security requirements belong to 'Identification and Authentication Control' which is a category from IEC62443-4-2 standard. Having in mind that the components of the Device Connect use case interact with open network, it is not surprising that the most of the security requirements belong to this category.

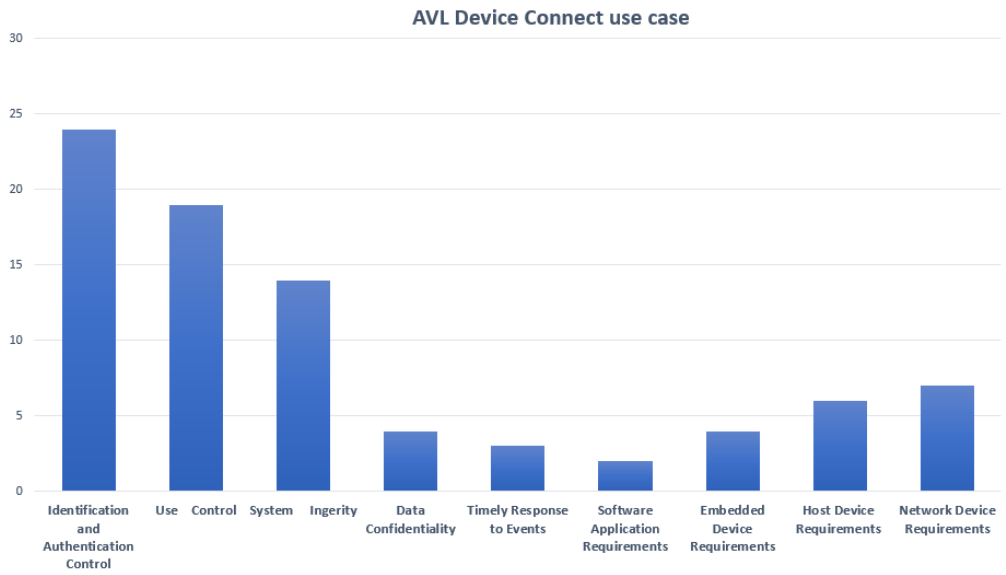

Figure 7: MoReTo Evaluation results for the Device Connect use case

5. Trusted Localization

Secure and accurate localization information is among the main challenges for autonomous driving and still an open problem for IIoT applications. In this section the main issues affecting current solutions are discussed, as well as existing potential concepts which can be used to overcome these issues. The main application scenario addressed in this section is about autonomous driving, however, the developed concept is not restricted to this application. This section does not provide a full survey on the state-of-the-art localization approaches within IoT, but an overview on specific potentials technologies for device-based secure localization. Many of the existing localization attacks cannot be prevented by cryptographical means. Therefore, depending on the attack model, a specific technology, approach or method is required to enable secure localization. It is therefore necessary to take the potential attack vectors into consideration when selecting an appropriate localization technique to enable a secure localization of a given IoT system. We refer readers to [FAK19] for a detailed review on current localization approaches.

We start this section by introducing potential attack models, which need to be considered in the context of localization fraud. Then, we highlight key points and recent improvements in localization systems for IoT, as well as possible drawbacks and critical factors that should be considered in our work. Next, we consider the specific scenario of autonomous driving and its requirements towards secure localization techniques. A hardware-based localization approach, which allows to overcome some of the drawbacks of the existing approaches is presented. The section closes with remarks on what should be considered when designing a system and choosing adequate localization methods and technologies.

5.1 The System Model and the Localization Attack Model in IoT4CPS

In this section, the system model and the localization attack models are outlined. We consider systems of devices whose objectives are to wirelessly obtain their correct distance from other devices/objects/people. The system model differs regarding the technology that underlies the localization method. In the case of a ranging based localization, the system model consists of two or

more devices. At least one of the devices denotes a *verifier* (or set of verifiers), which aim is to estimate its distance to a *prover* using ranging methods. *While the verifier is trusted, and its location is known this does not hold for the prover.* With at least three *verifiers* it would be possible to estimate the 2-D localization of a *prover. In the case of LiDAR or camera-based localization the system model consists of a single device, which aims to estimate its distance from* an object/person/non-communicating device. In contrast to ranging, such methods do not require an explicit response to estimate the distance to other objects. In the case of GNSS the system model consists of a single device which aims for localizing itself.

The definitions regarding localization attack models are kept as general as possible and exclude attacks that are unrealistic in the considered application domains (e.g., physically blocking signals with metallic shields). Attackers are considered to have full control over a device's hardware. In addition, attackers can be dishonest participants of a protocol.

When considering ranging measurements, the following attacks are possible [D95]:

- Impersonation: An impersonation fraud is an attack where an adversary acting alone purports to be a legitimate prover.
- Distance Fraud: A distance fraud is an attack where a dishonest prover purports to be in the neighborhood of the *verifier*. He cheats without help of other entities located in the neighborhood.
- Mafia Fraud: A mafia fraud is an attack where an adversary defeats a distance-bounding protocol using a man-in-the-middle between the *verifier* and an honest prover located outside the neighborhood.
- Terrorist Fraud: A terrorist fraud is an attack where an adversary defeats a distance-bounding protocol using a man-in-the-middle between the *verifier* and a dishonest prover located outside of the neighborhood under the following circumstances. The dishonest prover actively helps the adversary to maximize her current attack success probability but without giving her any advantage for future man-in-the-middle attacks. (In such attacks, the man-in-the-middle (MiM) would attempt to pass the distance-bounding protocol as a valid prover/tag that the MiM does not represent/possess.)

Additional attacks which should be considered, as they are affecting localization methods such as LiDAR, GPS or camera-based approaches are:

- Jamming: means to transmit interference in the communication channel in order to overshadow another transmitter signal in the same channel.
- Spoofing: which is the injection of additional, non-genuine signals in the channel.

5.2 Secure Localization for Indoor Applications

In many IoT scenarios of smart homes, smart buildings or smart factory, indoor localization plays an important role. The main technologies that are currently in the focus of investigation, are WiFi, RFID, UWB and Bluetooth [FAK19].

WiFi-based localization systems are widely investigated in the literature. Due to hardware limitations, time-of-flight (ToF) techniques are usually not successful in providing decimeter level accuracy. Good accuracy can already be obtained with a AoA MIMO-based system, e.g., relying on a physically distributed infrastructure. Nonetheless, such a system is a typical target of jamming and spoofing attacks, which can make the system believe that the *prover* (unlocalized device) is in a completely different position. Also, in many scenarios, a dishonest *prover* can deceive such a system.

Fingerprinting-based system were also investigated within WiFi, and in this case, the resistance against attacks and the ability to detect them were considered [W13]. An attack can be detected and avoided, for instance, by selecting reliable signals that minimize the median of the distance.
Different systems and methods were also proposed for localization within RFID [SSA16]. A reader can localize itself, for instance, by using RSS measurements from active tags. This method is difficult to protect against attacks such as the *distance fraud,* in which a dishonest tag can change its signal at different power levels, thus disturbing the localization of the reader. Neither a solution nor an analysis method was found considering distance attacks against localization systems employing RFID at an indoor level. In contrast, privacy has been investigated in previous research and will therefore not be outlined here (the interested reader may refer to [A06, L09] for more information on ongoing privacy-related research in the context of RFID).
Besides featuring a (sub)decimeter accuracy and good performance against multipath, UWB-based systems can offer good security levels if using a Two-Way-Ranging (TWR) approach by incorporating distance bounding as a building block. The authors in [M17] design and analyse a new protocol that is resistant to the most popular attacks. It is still vulnerable against the Distance Enlargement Fraud, for instance, which cannot be overcomed by any protocol relying on round-trip time measurements of propagating waves.
Most of the Bluetooth-based protocols for localization rely on RSSI measurements. The most popular example in this context is iBeacon. In [IMEC18], a recent commercial approach is presented using a combination of phase-based distance estimation with advanced signal processing, claiming to reach sub-decimeter accuracy using Bluetooth while being resistant against relay attacks. Also, the specification of the Bluetooth version 5.1 allows for AoA and AoD approaches by incorporating MIMO antenna arrays. This is expected to lead to better levels of security in the future.

5.3 Secure Localization for Autonomous Driving

The state-of-the-art localization technologies for autonomous driving include: mapping, GNSS receivers, network interfaces, radars, LiDARs, ultra-sound system, IMUs, cameras and fingerprinting [SSKMFA18].
GNSS receivers are the main technology for vehicular self-localization, which can provide a global positioning estimative at a low cost. Although the current accuracy of GPS may seem insufficient for the end users, a centimeter-level accuracy is expected to be achieved with the use of the combined GPS-Galileo system [SSKMFA18], which means sufficient accuracy to keep a vehicle in lane. The main dependability issue of GNSS to autonomous vehicles is the lack of availability when those are driving through urban environments or other environments with obstacles which can block the incoming signals from satellites, such as tunnels and bridges. The attempt to combine GPS with IMU sensors has shown significant improvement in recent research, but still not enough for the autonomous driving. Additionally, they are still subject to spoofing attacks, such as the one shown in [S17]. The solution proposed in the same paper disregards sophisticated attackers and is thus unrealistic. An extensive analysis of the vulnerabilities of GNSS systems can be found in [RTG16].
Camera-based systems can offer resolution and sharpness better than the human eye. The main challenges within this technology are specially in the data processing in order to select important aspects of a scene [BBBGP09]. Similar to the human sight, the main drawback of this technology is the strong impact of lighting conditions, even for far-infrared cameras in low-visibility scenarios. These drawbacks create a security gap for attacks that can be performed even by an attacker with a

low level of sophistication [PJ15], such as blinding the camera. Redundancy, i.e., using multiple cameras, and adding optical filters are the two main solutions for this attack, but can still not prevent a more sophisticated attack. Attacks are also possible for Lidar [BBBGP09, SKKK17], ultra-sound systems [JJY09], as well as for the other technologies already mentioned.

Another approach is to make use of a trusted infrastructure and V2I communication [SJ06, JSJ04, D95]. In such a scenario, different ranging measurements between two devices can be obtained using time-of-flight techniques. To compute the localization of an unlocalized device, a triangulation of (at-least three) ranging estimations from different devices can be used. This approach avoids most of the current attacks, including the Distance Enlargement Fraud (DEF), as it relies on proofed distance bounding techniques. Research on distance bounding has already covered all the known attacks [GMIS18] in this area and focused mostly on distance reduction attacks, i.e., attacks in which the attacker aims to convince the *verifier* that the attacker or another *prover* is closer to the verifier than he actually is. The *verifier*, in this context, is the vehicle or infrastructure device measuring its distance to the *prover*, an unlocalized device. The DEF can be covered by the approach in [6], as far as the vehicle is within the region/triangle involved by at least three verifiers in the 2-D case. In the case of direct distance measurements between two vehicles (V2V) using two-way ranging of electromagnetic waves, it was proven that an attacker can always succeed in the DEF [ZSA15]. This scenario is also important as the measurements can be used to construct an overall map, which includes other vehicles, objects, etc.

5.4 Proposed Secure Localization Concept

In order to develop a concept that can serve as a basis of a secure localization system, that would be capable of overcoming the DEF, we investigate the potential of coupling mechanisms for localization, e.g. for ranging between two devices. The proposed approach is similar to distance bounding protocols, and relies on the fact that electromagnetic fields propagate at the speed of light. The critical difference is that in the traditional two-way ranging of electromagnetic waves techniques use the propagating waves, while in the proposed approach, the transmitted fields remain coupled to the transmitter/verifier. The transmitter/verifier can sense the moment when the transmitted field reaches the receiver/*prover*. In other words, the *prover* can only receive a signal if it is coupled to the *verifier*, thus disturbing the transmitted field. This disturbance can be sensed by the *verifier* and, therefore, be used to detect possible frauds. Although coupling mechanisms have still a limited communication range, they have recently gained special attention, and their communication range is continuously increasing at a fast pace.

We illustrate the concept using the inductive coupling use-case, and the same principle may extend to other coupling mechanisms. Considering a system comprising to spatially separated coils, namely primary coils *PC* and secondary coil *SC,* the goal of *PC* is to estimate its distance from *SC.* The devices to which the coils belong are entitled *verifier* and *prover*, respectively. Assuming that the magnetic field propagates at the speed of light (c), and that as soon as the field reaches *SC,* it will induce a current in SC, which will induce a current back in PC, the approach works as follows:

1. *V* will send a modulated nonce (random number) to *P* at t_V,sent, and start to monitor the voltage difference at its coils' terminals;
2. If *P* is coupled/tuned, it will receive the nonce after t_P_rec;
3. The reception of the nonce affects *V* after t_P_rec'. t_P_rec can be approximated by

$$t_{P,rec} = (t'_{P,rec} + t_{V,sent})/2$$

and the distance between *V* and *P* can be calculated by

$$d = (t_{P,rec} - t_{V,sent}) . c$$

4. *P* sends the nonce back to *V*. The calculated distance is correct and *P* is honest if and only if the received nonce matches the one sent.

Alternatively, d could be calculated as in the usual single-sided two-way ranging technique:

$$d = (t_{V,rec} - t_{V,sent} - \Delta_{Proc}). c/2$$

where Δ_{Proc} is P's processing time, which can be calculated by *V* in this approach.

The disturbance caused in *V* due to *P* is typically very small and distance dependent. A clear notion regarding their distance is visible when subtracting a reference curve, consisting of voltage measurements in *PC* without the *SC*, from the curve obtained when *SC* is present. Therefore, it is required to obtain a reference curve beforehand, i.e., a measurement without P. This step is called a "calibration phase".

Since this is a novel work and the physical mechanisms of the underlying principle cannot be directly implemented on any known off-the-shelf hardware, it was decided to perform a first evaluation based on numerical simulations. This allows an initial feasibility test before implementing the approach in real-world hardware.

There exist a range of electromagnetic simulators, which are capable to perform transient magnetic analysis. To the best of our knowledge, none of the available simulators allows to reproduce the finite propagation speed effects of the magnetic field. Due to the low frequency and small distances between coils in usual power applications, these effects are usually not considered [P00] and thus not required in simulations. This retardation should be considered if the primary and secondary coils are far apart and is critical in this approach. In order to overcome this issue, FEMM [D13], an open-source software capable of simulating static magnetics problems, was used in addition with a model specifying the propagation of the magnetic field at the simulation model level, namely the "expanding ring model", which avoids altering FEMM itself. The propagation model used facilitates the incorporation of the retarded potentials [O89] in FEMM. In order to validate the model, a different problem with known analytical solution was simulated and the resulting magnetic potential (A) was compared with the analytical one. The resulting plots are illustrated in Figure 8, and serve as an evidence that the used model approximates the equations for retarded potentials.

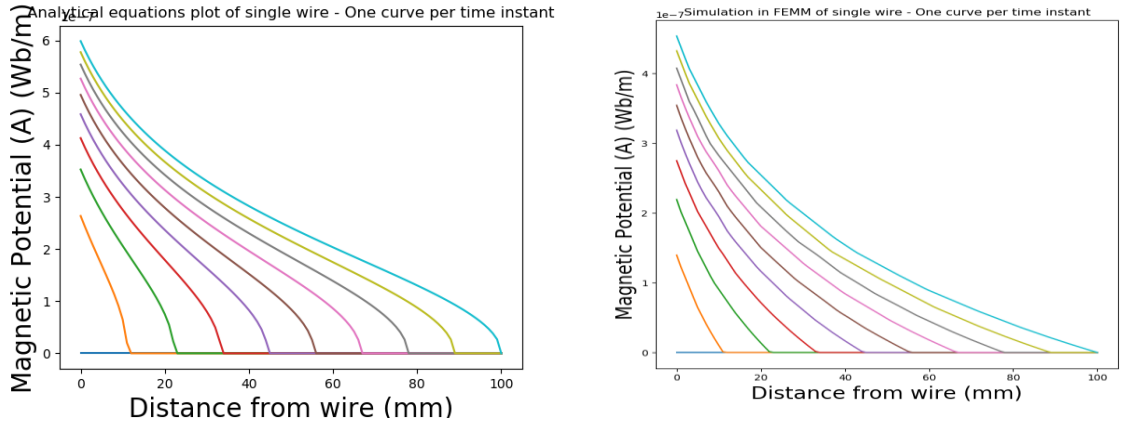

Figure 8: Magnetic potential over distance from PC measured for different time instants while the simulation boundary propagates.

The proposed underlying principle was thus simulated for different angles and distances between coils (reader and tag) and the voltage induced in *P* (the reader's coil or *PC*) was measured. An exemplary setup is shown in Figure 9 as a guideline for the reader willing to reproduce the experiment.

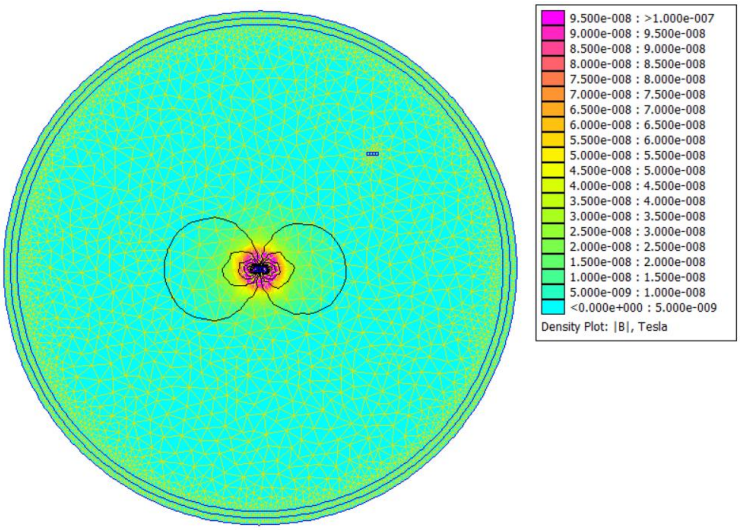

Figure 9: An exemplary setup

Also, some specific curves were plotted in Figure 9, after subtracting from the respective reference curves. Figure 10 shows the voltage measured in the primary with a sampling rate of 10.25 GSPS. Each column represents a specific distance between coils. The distance on the x-axis represents the distance propagated by the magnetic field. Secondary is oriented to the simulation's north. The results suggest that it is possible to establish the distances between the coils based on the RMS voltage, independent of the orientation of the coils.

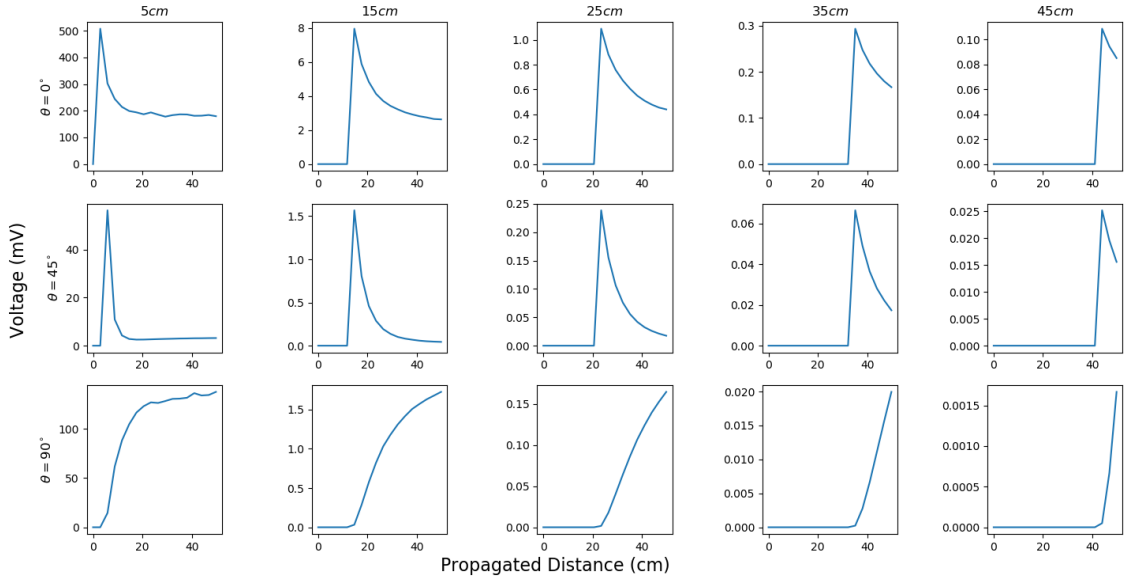

Figure 10: RMS voltage measured

When using a hardware with sampling frequency of 10.25 GSPS and a threshold of 1uV, the mean error of the distance is 1.21 cm when the *SC* is oriented to the north and 1.25 cm when *SC* is oriented to the outside of the simulation boundary. However, this resolution is not achievable with existing hardware. Considering a COTS ADC with a 12-bit resolution and 1.4 V analog input range, the mean

error of the distance increases to 8.14 cm when *SC* is oriented to the north and 13.02 cm when *SC* is oriented to the outside. Noise was not considering in the simulation and is also expected to reduce the accuracy of the system. Bandwidth is another limiting factor, since the required bandwidth to achieve a resolution of 2.93 cm is estimated to be in the order of 5.125 GHz.
Given this positive first evaluation, in a next step the approach will be implemented in a testbed to validate it with real-world devices. In addition, future research will focus on the question to what extent voltage difference signals could be used to infer the relative location of a tag towards an anchor, and to test the approach with multiple tags. Finally, the combination of the amplitude-level information with timing information could be investigated to improve the accuracy of the current approach and its level of security.

5.5 Recommendations for Secure Localization

Tabkle 2 summarizes the related technologies and potential attacks (still unsolved). Depending on specific use-cases attack models, the system designer should decide which technologies are applicable and which should be avoided.

Table 2: Recommendations for Secure Localization

Technology	Threats	Notes
UWB	DEF	DEF can be overcome within a triangle delimited by a trusted infrastructure.
WiFi	Mafia Fraud (untrusted infrastructure)	> 3m accuracy without attacker
Bluetooth	All but relay-attacks	Not considering Bluetooth 5.1
RFID	All	Privacy is a major concern
Camera-based	Jamming/blinding	No definitive countermeasure
LiDAR	Jamming/Saturating + Spoofing	Existing impractical solutions
GNSS	Jamming + Spoofing	Many solutions to spoofing exist but are not yet necessarily implemented in a system level.

6. Security Assurance through Threat Modelling, Security Analysis and Penetration Testing

Penetration tests are authorized attacks on a computer system performed with the aim to evaluate the security of a system. When evaluating the security of IoT systems with certain security assurance tools/penetration testing tools, we have to define the attacker model, the procedure of the penetration tests, as well as the information needed for the input parameters of the security assurance/penetration testing tools. Figure 11 gives an overview of the possible attack scenarios for different components of IoT/IIoT systems. More details about threat modelling, security analysis and penetration testing will be discussed in WP4 of the IoT4CPS project.

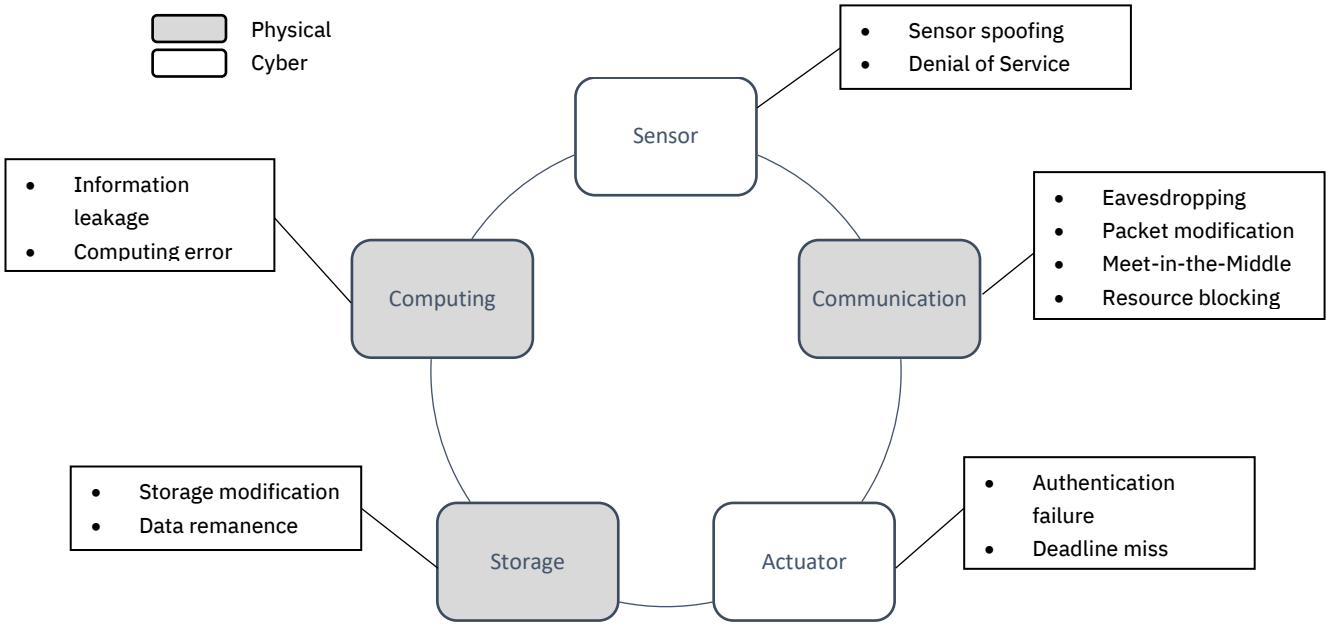

Figure 81 – Attacks on IIoT Systems

6.1.1 Target/Attack Model

An important consideration when defining the security requirements for IoT models and also when selecting the appropriate security assurance/penetration testing tools, is to consider an appropriate attack model. In general, we can define three attacker models, based on the information that the attacker has.

- Black box model: Only basic information about the target system/network is available, or no information is available.
- Grey box model: Limited information is available to the attacker (i.e. this can be knowledge about the target network, number of hosts, network infrastructure ...).
- White box model: Detailed background information is available to the attacker. Moreover, system information is available (i.e. the attacker has access to the source code, has a detailed overview of the network architecture, ...)

While the black box model is the most realistic model, as an attacker typically accesses the target system from an outside network (e.g. the Internet), an attacker however can also gain access within a network (i.e. by getting physical access). In that case, an attacker can scan the network and obtain valuable information that can later be used in an attack.

6.1.2 Penetration Testing Phases

The process of penetration tests can be simplified into the following phases.

- Reconnaissance: In this initial phase, information about the target system, software and users is gathered. This information can later be used to attack the target system.
- Scanning: In this phase, technical tools are used to further the attacker's knowledge about the target system.
- Gaining Access: In this phase, the attacker actively tries to exploit the systems by using the information gathered in the previous phases.
- Maintaining Access: An attacker is required to persistently be able to access the target system to gather as much data as possible.

- Covering Tracks: An attacker must erase any traces, such as log files, in order to remain anonymous.

6.1.3 Necessary Input Information

We analysed several penetration testing tools (most of them are freely available in Kahli Linux[O19]) regarding the input requirements they need, to optimally obtain as much information as possible from a target, as well as the information needed to find targeted vulnerabilities, and exploits. We arrange the input requirements for the penetration testing tool according to the penetration testing phases as defined above. **Table 1 gives an overview of the necessary input information for security assurance/penetration testing tools. We just list the first three penetration testing phases, as they can easily be automated. Using more detailed exploits and maintaining access to a target, often requires user interaction.**

Table 3: Required input parameters for penetration testing tools

Penetration Testing Phase	Input Requirements
Reconnaissance	IP addresses, host names, network addresses, network interfaces
Scanning	Web URL's, IP addresses, network interfaces, port numbers, operating system information
Gaining Access	Operating system information, software versions, protocol versions

6.2 Security Requirements for IoT Components/WP3 Models of IoT4CPS

In the following, we will give a non-exhaustive list of security requirements that the models of IoT/IIoT components should include, to optimise the automated information gathering phase of security assurance/penetration testing tools. In general, the more information that can be abstracted from the real world into a model, the more information can then be used in the attacks. Additionally, we also have to consider different attacker models, to model the capabilities of an attacker. For simplifications, we consider the Dolev-Yao attacker model *[DY83]*, which means that an attacker can overhear, intercept and synthesize any message in the network, and is only limited by the cryptographic methods that are used. Moreover, an attacker can encrypt and decrypt with any keys she knows/successfully obtains.

6.2.1 Limitations and Restrictions

Depending on the size of the IoT system a detailed security analysis and detailed penetration tests can quickly require many resources. Therefore, we have to limit the level of detail of the security analysis and take some assumptions into consideration. In general, we assume that the cryptographic primitives are based on well-studied cryptographic standards and therefore we do not consider detailed attacks on the building blocks (i.e. such as block ciphers ...). This does not include misconfigurations of cipher suites, as that are one of the most commonly errors. Moreover, not all steps in penetration tests can be automated. Therefore, some penetration tests have to be managed by experienced security auditors and cryptographers.

6.2.2 Required properties of IoT/IIoT components for safe and secure IoT/IIoT models

In the following, we list desired parameters that should be considered in the IoT/IIoT models of WP3. If some parameters are unknown in the initial modelling phase, they can be added later during an information gathering phase, which is part of penetration testing, or be specified when a more detailed threat model is generated. Moreover, the missing parameters can also be added with the IoT device detection tool that will be developed as part of WP4.

Smart Devices and Sensors: Smart devices and sensors that are connected to networks are potential threats as they can have security vulnerabilities due to outdated operating systems, unsecure software versions, open network ports, unsecured hardware interfaces, and many more security issues [KKSV17]**. The models that represent the smart devices and sensor components should therefore include the parameters as listed in Table 4 shown below.**

Table 4: Desired parameters of smart devices and sensor components

Parameter	Description	Impact/Priority
Hardware interface	Lists the available interfaces of the smart device/sensor (i.e. network, USB ...).	High
IP address	Indicates the IP address of the smart device/sensor.	High
MAC address	Indicates the MAC address of the smart device/sensor.	High
Operating system	Indicates the operating system used by the smart device/sensor.	High
Firmware version	Indicates the firmware that is used by the smart device/sensor.	High
Network protocols	Lists the available supported network protocols.	High
Secure key store	Indicates if the smart devices/sensor has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the smart devices/sensor has internal data storage.	High
Power consumption	Indicates the power consumption of the smart device/sensor.	Medium
Electromagnetic emission	Indicates the electromagnetic emission of the smart device/sensor.	Medium
Pairing process	Indicates how the smart device/sensor can connect to other devices.	Medium
Update process	Indicated how the smart device/sensor receives software/firmware updates (i.e. over the air, ...).	Medium
Reset functionality	Indicates how the smart device/sensor can be reset to an initial setting.	Medium

Networks and Protocols: **IoT networks are used to interconnect several IoT devices and sensors with backend systems or the cloud via the Internet. Often different types of networks are used, also with many different protocols. Potential threats in IoT networks can occur, if outdated protocols are used that have security vulnerabilities** [BBD+15, MDK14, O14]**, or if different networks have different security requirements. Table 5 provides a list of desired parameters for the models that represent network and protocol components.**

Table 5: Desired parameters of network and protocol components

Parameter	Description	Impact/Priority
Connection type	Indicates if the network is a physical network (Ethernet) or if the network is wireless (WIFI, Bluetooth, NFC ...).	High
Protocol version	Describes the protocol version used (i.e. TLS 1.3).	High
Encrypted connection	Indicates if the connection is encrypted.	High
Data integrity provided	Indicates if the data integrity of the connection is ensured.	High
Source authenticated	Indicates if the source is authenticated.	High
Destination authenticated	Indicates if the destination is authenticated.	High
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Throughput	Indicates the actual rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Latency	Indicated the delay between a data packet sent from the source until it is received at the destination.	Medium
Error rate	Indicates the number of corrupted bits as a percentage of the total sent bits.	Medium

Gateways: **IoT gateways are physical devices or software programs that serve as a connection point between smart devices/sensors and servers in the cloud. Those gateways are used to perform protocol translation, data processing, data storage and filtering. Potential threats can occur in the protocol translation when different security requirements are set or unsecured network interfaces, open hardware ports or vulnerable software is used. Table 6 lists the desired parameters for the models that represent gateway components.**

Table 6: Desired parameters of gateway components

Parameter	Description	Impact/Priority
Network interface	Lists the available interfaces of the IoT gateway.	High

IP address	Indicates the IP address of the IoT gateway.	High
MAC address	Indicates the MAC address of the IoT gateway.	High
Operating system	Indicates the operating system used by the IoT gateway.	High
Firmware version	Indicates the firmware that is used by the IoT gateway.	High
Network protocols	Lists the available supported network protocols.	High
Secure key store	Indicates if the IoT gateway has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the IoT gateway has internal data storage.	High
Pairing process	Indicates how the gateway can connect to other devices.	Medium
Update process	Indicated how the gateway receives software/firmware updates (i.e. over the air, ...).	Medium
Reset functionality	Indicates how the gateway can be reset to an initial setting.	Medium
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium

Cloud/Servers: The cloud is often used in IoT systems for infrastructure, for data storage and for data processing and analytics. Cloud-based services are usually provided by third parties. Therefore, potential threats can occur if data is not encrypted, or an adversary has access to some servers in the cloud. Table 7 lists the desired parameters for the models that represent cloud services and server components.

Table 7: Desired parameters of Cloud/Server components

Parameter	Description	Impact/Priority
Network interface	Lists the available interfaces of the cloud server.	High
IP address	Indicates the IP address of the cloud server.	High
MAC address	Indicates the MAC address of the cloud server.	High
Web URL	Indicates the URL of the web interface of the cloud service.	High
Operating system	Indicates the operating system used by the cloud server.	High
Network protocols	Lists the available supported network protocols.	High

Secure key store	Indicates if the cloud/server has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the cloud/server has internal data storage.	High
Software versions	Indicates the software versions running on the cloud/server.	High
Shared Resources	Indicates if the cloud server shares resources with other users.	High
Software APIs	Lists the available APIs that can be used to interact with the cloud/server.	Medium
Server CPU	Indicates the CPUs used by the servers in the cloud.	Medium
Server controllers	Indicates the controllers (i.e. USB, hard drive, media) used by the servers in the cloud.	Medium
Server interfaces	Indicated the interface (i.e. network, USB…) used by the server in the cloud.	Medium

Smart Service and Backend Systems: Often IIoT devices and sensors in smart factories produce huge amounts of data that are first filtered and pre-processed in the cloud, and then further processed and stored in backend systems within a company network. Potential threats for those smart services and backend systems can occur from denial of service (DoS) attacks, as well as through open network ports, outdated software, misconfigured hardware/software and other security issues. Table 8 lists the desired parameters for the models that represent smart services and backend system components.

Table 8: Desired parameters of smart services and backend system components

Parameter	Description	Impact/Priority
Network interface	Lists the available interfaces of the smart service/backend system.	High
IP address	Indicates the IP address of the smart service/backend system.	High
MAC address	Indicates the MAC address of the smart service/backend system.	High
Operating system	Indicates the operating system used by the smart service/backend system.	High
Network protocols	Lists the available supported network protocols.	High
Secure key store	Indicates if the smart service/backend system has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High

Data storage	Indicates if the smart service/backend system has internal data storage.	High
Software versions	Indicates the software versions running on the smart service/backend system.	High
Software APIs	Lists the available APIs that can be used to interact with the smart service/backend system.	Medium
CPU	Indicates the CPUs used by the smart service and backend system.	Medium
Controllers	Indicates the controllers (i.e. USB, hard drive, media) used by the smart service and backend system.	Medium
Hardware interfaces	Indicated the interface (i.e. network, USB…) used by the smart service and backend system.	Medium

6.3 Example according to AVL industrial device connect use case

In the following, we provide an example for the extraction of security relevant properties of WP3 models that can then be used in the WP4 security assurance tools according to the AVL industrial device connect use case.

6.3.1 Description of the AVL industrial Device.Connect use case

The AVL device connect use case can be used to showcase a typical scenario for an IIoT system. Figure 12 gives a detailed overview of the IIoT system under test.

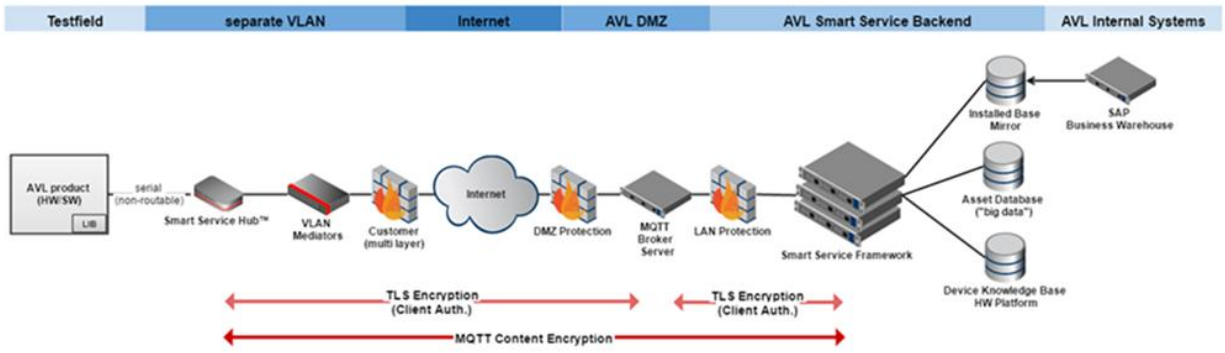

Figure 12: AVL Device Connect use case

When we want to abstract the use case into an abstract model we can identify the following architectural components:

- Smart Device and Sensors: The AVL product in Figure 2 could be a simple device, but could also be complex, intelligent measurement devices including docents of sensors and a sizeable computing platform.
- Gateways: Figure 12 shows an IoT gateway that includes the Smart Service Hub, the VLAN Mediator and the Customer Firewall. It is used to connect the AVL product to the Internet.

- Networks and Protocols: Figure 12 lists several networks. The first is a non-routable interface which connects the AVL product with the Smart Service Hub. Next, either a VLAN or Ethernet interface is used to connect the Smart Service Hub to the Internet. The Internet then connects the client side AVL product with the backend at the company network from AVL. Within the AVL company network a demilitarized zone (DMZ) is used to separate the company network from the internal network. In all those networks, several protocols are used, including a serial (non-routable) protocol, the TLS protocol and MQTT.
- **Cloud/Servers: In the Device Connect use case, no cloud-based services are used. However, the DMZ uses an MQTT Broker server. In general, we could also model the servers used in the Internet, but we disregard it here for the sake of brevity.**
- **Smart Service and Backend Systems: Figure 12 shows the AVL Smart Service Backend that connects a Smart Service Framework, consisting of several databases (Installed Base Mirror, Asset Database, and Device Knowledge Base), via the DMZ, using an MQTT Broker, to the Internet that connects several AVL products that are running at different clients.**

6.3.2 Security Requirements for the architectural models of the AVL device connect use case

To optimise the use of security assurance tools, we now try to specify the necessary parameters that the architectural models, as defined above for the AVL device connect use case, should include.

- **Smart Devices and Sensors: Let's assume for simplicity that the AVL product consists as one single simple device. The desired parameters for the security assurance tools are defined in Table 9.**

Table 9: Desired parameters of the AVL product

Parameter	Description	Impact/Priority
Hardware interfaces	Lists the available interfaces of the AVL product.	High
Operating system	Indicates the operating system used by the AVL product.	High
Firmware	Indicates the firmware used by the AVL product.	High
Network protocols	Lists the available supported network protocols.	High
Secure key store	Indicates if the AVL product has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the AVL product has internal data storage.	High
Software versions	Indicates the software versions running on the AVL product.	High
Software APIs	Lists the available APIs that can be used to interact with the AVL product.	High

Power consumption	Indicates the power consumption of the AVL product.	Medium
Electromagnetic emission	Indicates the electromagnetic emission of the AVL product.	Medium
Pairing process	Indicates how the AVL product can connect to other devices.	Medium
Update process	Indicated how the AVL product receives software/firmware updates (i.e. over the air, ...).	Medium
Reset functionality	Indicates how the AVL product can be reset to an initial setting.	Medium

- **Gateways: The Smart Service Hub is the IoT gateway that is used to connect the AVL product to the Internet. Table 10 lists the desired parameters for the security assurance tools.**

Table 10: Desired parameters of the Smart Service Hub

Parameter	Description	Impact/Priority
Hardware interfaces	Lists the available interfaces of the Smart Service Hub.	High
Operating system	Indicates the operating system used by the Smart Service Hub.	High
Firmware	Indicates the firmware used by the Smart Service Hub.	High
Network protocols	Lists the available supported network protocols.	High
Network ports	Lists the open network ports of the Smart Service Hub.	High
Secure key store	Indicates if the Smart Service Hub has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the Smart Service Hub has internal data storage.	High
Pairing process	Indicates how the smart service hub can connect to other devices.	Medium
Update process	Indicated how the smart service hub receives software/firmware updates (i.e. over the air ...).	Medium
Reset functionality	Indicates how the smart service hub can be reset to an initial setting.	Medium
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium

- Networks and Protocols: The AVL Device Connect use case uses several networks. The first network, let's call it N1, connects the AVL product with the Smart Service Hub. Table 11 **lists the desired parameters for the security assurance tools for network N1.**

Table 11: Desired parameters of the network N1

Parameter	Description	Impact/Priority
Connection type	Indicates if the network N1 is a physical network (Ethernet) or if the network is wireless (WIFI, Bluetooth, NFC ...).	High
Protocol version	Describes the protocol version used in network N1 (i.e. TLS 1.3).	High
Encrypted connection	Indicates if the connection is encrypted.	High
Data integrity provided	Indicates if the data integrity of the connection is ensured.	High
Source authenticated	Indicates if the source is authenticated.	High
Destination authenticated	Indicates if the destination is authenticated.	High
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Throughput	Indicates the actual rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Latency	Indicated the delay between a data packet sent from the source until it is received at the destination.	Medium
Error rate	Indicates the number of corrupted bits as a percentage of the total sent bits.	Medium

The second network, let's call it N2, connects the Smart Service Hub to the Internet. Table 12 **lists the desired parameters for the security assurance tools for network N2.**

Table 12: Desired parameters of the network N2

Parameter	Description	Impact/Priority
Connection type	Indicates if the network N2 is a physical network (Ethernet) or if the network is wireless (WIFI, Bluetooth, NFC ...).	High
Protocol version	Describes the protocol version used in network N2 (i.e. TLS 1.3).	High
Encrypted connection	Indicates if the connection is encrypted.	High
Data integrity provided	Indicates if the data integrity of the connection is ensured.	High

Source authenticated	Indicates if the source is authenticated.	High
Destination authenticated	Indicates if the destination is authenticated.	High
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Throughput	Indicates the actual rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Latency	Indicated the delay between a data packet sent from the source until it is received at the destination.	Medium
Error rate	Indicates the number of corrupted bits as a percentage of the total sent bits.	Medium

In general, we could also model the Internet as a network. However, for simplicity, let's disregard the Internet in our analysis here, as we generally do not have control over it.
The AVL company network is made up of three networks in the use case. The first network, let's call it N3, connects the Internet with a DMZ. The second network is the DMZ that runs a MQTT Broker Server. The third network, let's call it N4, is the internal AVL network that combines a Smart Service Framework, several databases and a SAP Business Warehouse. Tables 13, 14 and 15 **list the desired parameters for the security assurance tools for the networks N3, DMZ and N4, respectively.**

Table 13: Desired parameters of the network N3

Parameter	Description	Impact/Priority
Connection type	Indicates if the network N3 is a physical network (Ethernet) or if the network is wireless (WIFI, Bluetooth, NFC ...).	High
Protocol version	Describes the protocol version used in network N3 (i.e. TLS 1.3).	High
Encrypted connection	Indicates if the connection is encrypted.	High
Data integrity provided	Indicates if the data integrity of the connection is ensured.	High
Source authenticated	Indicates if the source is authenticated.	High
Destination authenticated	Indicates if the destination is authenticated.	High
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium

Throughput	Indicates the actual rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Latency	Indicated the delay between a data packet sent from the source until it is received at the destination.	Medium
Error rate	Indicates the number of corrupted bits as a percentage of the total sent bits.	Medium

Table 14: Desired parameters of the network DMZ

Parameter	Description	Impact/Priority
Connection type	Indicates if the network DMZ is a physical network (Ethernet) or if the network is wireless (WIFI, Bluetooth, NFC ...).	High
Protocol version	Describes the protocol version used in network DMZ (i.e. TLS 1.3).	High
Encrypted connection	Indicates if the connection is encrypted.	High
Data integrity provided	Indicates if the data integrity of the connection is ensured.	High
Source authenticated	Indicates if the source is authenticated.	High
Destination authenticated	Indicates if the destination is authenticated.	High
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Throughput	Indicates the actual rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Latency	Indicated the delay between a data packet sent from the source until it is received at the destination.	Medium
Error rate	Indicates the number of corrupted bits as a percentage of the total sent bits.	Medium

Table 15: Desired parameters of the network N4

Parameter	Description	Impact/Priority
Connection type	Indicates if the network N4 is a physical network (Ethernet) or if the network is wireless (WIFI, Bluetooth, NFC ...).	High
Protocol version	Describes the protocol version used in network N4 (i.e. TLS 1.3).	High

Encrypted connection	Indicates if the connection is encrypted.	High
Data integrity provided	Indicates if the data integrity of the connection is ensured.	High
Source authenticated	Indicates if the source is authenticated.	High
Destination authenticated	Indicates if the destination is authenticated.	High
Bandwidth	Indicates the maximum rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Throughput	Indicates the actual rate of network packets that can be sent over a time interval (usually measured in bits/second).	Medium
Latency	Indicated the delay between a data packet sent from the source until it is received at the destination.	Medium
Error rate	Indicates the number of corrupted bits as a percentage of the total sent bits.	Medium

- **Cloud/Servers: The DMZ in the AVL network offers a MQTT Broker server.** Table 16 **lists the desired parameters for the security assurance tools for the MQTT Broker server.**

Table 16: Desired parameters of the MQTT Broker server

Parameter	Description	Impact/Priority
Network interface	Lists the available interfaces of the MQTT Broker server.	High
IP address	Indicates the IP address of the MQTT Broker server.	High
MAC address	Indicates the MAC address of the MQTT Broker server.	High
Operating system	Indicates the operating system used by the MQTT Broker server.	High
Network protocols	Lists the available supported network protocols.	High
Secure key store	Indicates if the MQTT Broker server has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the MQTT Broker server has internal data storage.	High
Software versions	Indicates the software versions running on the MQTT Broker server.	High

Shared Resources	Indicates if the MQTT Broker server shares resources with other users.	High
Software APIs	Lists the available APIs that can be used to interact with the MQTT Broker server.	Medium
Server CPU	Indicates the CPUs used by the MQTT Broker server.	Medium
Server controllers	Indicates the controllers (i.e. USB, hard drive, media) used by the MQTT Broker server.	Medium
Server interfaces	Indicated the interface (i.e. network, USB…) used by the MQTT Broker server.	Medium

- **Smart Service and Backend Systems: The AVL Smart Service Backend consists of a Smart Service Framework that connects to several databases (Installed Base Mirror, Asset Database, and Device Knowledge Base) as well as to an internal system (SAP Business Warehouse).** Table 17 **lists the desired parameters for the security assurance tools for the Smart Service Backend.**

Table 17: Desired parameters of the AVL Smart Service Backend

Parameter	Description	Impact/Priority
Network interface	Lists the available interfaces of the AVL smart service backend system.	High
IP address	Indicates the IP address of the AVL smart service backend system.	High
MAC address	Indicates the MAC address of the AVL smart service backend system.	High
Operating system	Indicates the operating system used by the AVL smart service backend system.	High
Network protocols	Lists the available supported network protocols.	High
Secure key store	Indicates if the AVL smart service backend system has access to a secure key store (i.e. Secure Element, Hardware Security Module, or Trusted Platform Module).	High
Data storage	Indicates if the AVL smart service backend system has internal data storage.	High
Software versions	Indicates the software versions running on the AVL smart service backend system.	High
Software APIs	Lists the available APIs that can be used to interact with the AVL smart service backend system.	Medium
CPU	Indicates the CPUs used by the AVL smart service backend system.	Medium
Controllers	Indicates the controllers (i.e. USB, hard drive, media) used by the AVL smart service backend system.	Medium

Hardware interfaces	Indicated the interface (i.e. network, USB…) used by the AVL smart service backend system.	Medium

7. Conclusion

Since IoT is more and more entering industrial domains, such as factory automoation or automotive, the requirements on IoT safety and security are getting crucial. For example, on the automotive side, cars are getting more and more connected and driving is expected to become fully automated. On the side of industrial automation, an increasing number of industrial devices are wirelessly interconnected, some of them directly to the Internet. Thus, new concepts strongly addressing safety and security for IoT/IIoT are necessary. For security analysis, the security requirements tool called Model-Based Security Requirement Management Tool (MoReTo) was applied to the industrial device, AVL Device Connect. MoReTo copes with the ambiguity of understanding the system security requirements by managing a massive number of security countermeasures and determines the security requirements needed to achieve a high degree of security assurance. For trusted localization, a concept similar to distance bounding protocols is presented that relies on the fact that electromagnetic fields propagate at the speed of light. These and other conpcets presented in this report will help to decrease the engineering costs of the increasingly complex IoT systems.

8. References

[A06] A. Juels, "RFID security and privacy: a research survey," in IEEE Journal on Selected Areas in Communications, vol. 24, no. 2, pp. 381-394, Feb. 2006. doi: 10.1109/JSAC.2005.86139

[B19] Bluetooth Core Specification V5.1, Bluetooth SIG Proprietary, Jan. 2019

[BBBGP09] Bertozzi M., Bombini L., Broggi A., Grisleri P., Porta P.P. (2009) Camera-Based Automotive Systems. In: Belbachir A. (eds) Smart Cameras. Springer, Boston, MA. 2009.

[BBD+15] Beurdouche, B., Bhargavan, K., Delignat-Lauvaud, A., Fournet, C., Kohlweiss, M., Pironti, A., Strub, P., Zinzindohoue, J.K., A Messy State of the Union: Taming the Composite State Machines of TLS, May. 2015

[BG14] MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta.29 October2014. OASIS Standard. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.

[D13] Ph.D. David Meeker. FEMM 4.2 Magnetostatic Tutorial. 2013.

[D95] Deka, Bhaswati, "Secure Localization Topology and Methodology for a Dedicated Automated Highway System" (2013). All Graduate Theses and Dissertations. 1995.

[DH98] Deering, S., Hinden, R., Internet Protocol, Version 6 (IPv6) Specification, RFC2460, Dec. 1998

[DY83] Dolev, D., Yao, A. C. On the security of public key protocols, IEEE Transactions on Information Theory, IT-29 (2): 198–208, 1983

[FAK19] Faheem Zafari, Athanasios Gkelias, Kin Leung. A Survey of Indoor Localization Systems and Technologies. 2019.

[GMIS18] Gildas Avoine, Muhammed Ali Bingöl, Ioana Boureanu, Srdjan čapkun, Gerhard Hancke, Süleyman Kardaş, Chong Hee Kim, Cédric Lauradoux, Benjamin Martin, Jorge Munilla, Alberto Peinado, Kasper Bonne Rasmussen, Dave Singelée, Aslan Tchamkerten, Rolando Trujillo-Rasua, and Serge Vaudenay. 2018. Security of Distance-Bounding: A Survey. ACM Comput. Surv. 51, 5, Article 94 (September 2018), 33 pages.

[Hoeftberger2015] Oliver Höftberger. Knowledge-Based Dynamic Reconfiguration for Embedded Real-Time Systems. PhD thesis, TU Wien, Institute of Computer Engineering, Wien, 2015.

[IEC17] IEC 62443-4-2, “Industrial communication networks - Security for industrial automation and control systems - part 4-2 Technical security requirements for IACS components,” International Electrotechnical Commission, Tech. Rep., 2017.

[IMEC18]https://www.imec-int.com/drupal/sites/default/files/2018-11/Accurate%20and%20secure%20Distance%20Measurement%20with%20Bluetooth_0.pdf

[JJY09] Jerry T. Chiang, Jason J. Haas, and Yih-Chun Hu. 2009. Secure and precise location verification using distance bounding and simultaneous multilateration. In Proceedings of the second ACM conference on Wireless network security (WiSec '09). ACM, New York, NY, USA, 181-192

[JSJ04] J. P. Hubaux, S. Capkun and Jun Luo, "The security and privacy of smart vehicles," in IEEE Security & Privacy, vol. 2, no. 3, pp. 49-55, May-June 2004.

[KKSV17] Kolias, C., Kambourakis, G., Savrou, A., Voas, J., DDoS in the IoT: Mirai and Other Botnets, IEEE Computer Volume 50, Issue 7, 2017

[KMLS2017] R. Khan, K. McLaughlin, D. Laverty, and S. Sezer, "STRIDE-based threat modeling for cyber-physical systems", in 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), pages 1-6, Sept 2017.

[L09] Langheinrich, M. Pers Ubiquit Comput (2009) 13: 413. https://doi.org/10.1007/s00779-008-0213-4

[L17] LoRaWANTM 1.1 Specification, LoRa Alliance Technical Committee, Oct. 2017

[M17] Miri, J. "Secure Distance Bounding Protocol on Ultra-WideBand Based Mapping Code". World Academy of Science, Engineering and Technology, International Science Index 125, International Journal of Computer, Electrical, Automation, Control and Information Engineering, (2017), 11(5), 590 - 596.

[MDK14] Möller, B., Duong, T., Kotowicz, K., This POODLE Bites: Exploiting The SSL 3.0 Fallback, https://www.openssl.org/~bodo/ssl-poodle.pdf, Sep. 2014

[O14] OpenSSL, OpenSSL ‘Heartbleed’ vulnerability (CVE-2014-0160), Apr. 2014

[O19] Offensive Security, Kahli Linux Penetration Testing Tools, https://tools.kali.org/, 2019

[O89] O.D. Jefimenko. Electricity and Magnetism: An Introduction to the Theory of Electric and Magnetic Fields. Electret Scientific Company, 1989.

[P00] Pr Holmberg. Modelling the transient response of windings, laminated steel cores and electromagnetic power devices by means of lumped circuits : With special reference to windings with a coaxial insulation system. 01 2000.

[PJ15] Petit, Jonathan et al. “Remote Attacks on Automated Vehicles Sensors : Experiments on Camera and LiDAR.” 2015.

[RHISG2017] D. Ratasich, O. Höftberger, H. Isakovic, M. Shafique, and R. Grosu. A Self-Healing Framework for Building Resilient Cyber-Physical Systems. In 2017 IEEE 20th International Symposium on Real-Time Distributed Computing (ISORC), pages 133-140, May 2017.

[RHISG2017] D. Ratasich, O. Höftberger, H. Isakovic, M. Shafique, and R. Grosu. A Self-Healing Framework for Building Resilient Cyber-Physical Systems. In 2017 IEEE 20th International Symposium on Real-Time Distributed Computing (ISORC), pages 133-140, May 2017.

[RPGB2019] D. Ratasich, M. Platzer, R. Grosu, E. Bartocci, “Adaptive Fault Detection exploiting Redundancy with Uncertainties in Space and Time,” arXiv:1903.04326 [cs], Mar. 2019.

[RPSG2018] D. Ratasich, T. Preindl, K. Selyunin, and R. Grosu. Self-healing by property-guided structural adaptation. In 2018 IEEE Industrial Cyber-Physical Systems (ICPS), pages 199-205, May 2018.

[RTG16] R. T. Ioannides, T. Pany and G. Gibbons, "Known Vulnerabilities of Global Navigation Satellite Systems, Status, and Potential Mitigation Techniques," in Proceedings of the IEEE, vol. 104, no. 6, pp. 1174-1194, June 2016

[S17] S. Tayeb et al., "Securing the positioning signals of autonomous vehicles," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, 2017, pp. 4522-4528.

[S19] Z-Wave Plus Device Type v2 Specification, Silicon Labs, Mar 2019

[shsa-prolog] D. Ratasich. Implementation of SHSA in Prolog/ProbLog. GitHub Repository. Retrieved 2019-04-02 from https://github.com/dratasich/shsa-prolog.

[SJ06] S. Capkun and J. -. Hubaux, "Secure positioning in wireless networks," in IEEE Journal on Selected Areas in Communications, vol. 24, no. 2, pp. 221-232, Feb. 2006.

[SKKK17] Shin H., Kim D., Kwon Y., Kim Y. (2017) Illusion and Dazzle: Adversarial Optical Channel Exploits Against Lidars for Automotive Applications. In: Fischer W., Homma N. (eds) Cryptographic Hardware and Embedded Systems – CHES 2017. CHES 2017. Lecture Notes in Computer Science, vol 10529. Springer, Cham.

[SSA16] Singh, Shikha and Arun Aggarwal. “Survey on Localization Techniques of RFID for IOT.” (2016).

[SSKMFA18] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough and A. Mouzakitis, "A Survey of the State-of-the-Art Localization Techniques and Their Potentials for Autonomous Vehicle Applications," in IEEE Internet of Things Journal, vol. 5, no. 2, pp. 829-846, April 2018.

[ST19] Statista, https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, Accessed: 23.04.2019

[STRIDE] Microsoft Corporation. The STRIDE Threat Model. https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20). 2009.

[SYDZ16] Shang, W., Yu, Y., Droms, R., Zhang, L., Challenges in IoT networking via TCP/IP architecture. NDN, Technical Report NDN-0038, 2016.

[W13] Wei-Chia Lai et al., "A survey of secure fingerprinting localization in wireless local area networks," 2013 International Conference on Machine Learning and Cybernetics, Tianjin, 2013, pp. 1413-1417.

[Z12] ZigBee Specification, 053474r20, ZigBee Alliance, Sep 2012

[ZSA15] Zheng X., Safavi-Naini R., Ahmadi H. (2015) Distance Lower Bounding. In: Hui L., Qing S., Shi E., Yiu S. (eds) Information and Communications Security. ICICS 2014. Lecture Notes in Computer Science, vol 8958. Springer, Cham.