Secured and Easy-to-Use NFC-Based Device Configuration for the Internet of Things

Thomas Ulz, *Member, IEEE,* Thomas Pieber, *Member, IEEE,* Andrea Höller, *Member, IEEE*
Sarah Haas, *Member, IEEE,* and Christian Steger, *Member, IEEE,*

***Abstract*—Public awareness regarding security aspects in the Internet of Things (IoT) is currently rising due to regular media presence of various IoT-related security breaches. One of the major weaknesses of IoT devices is the absence of appropriate mechanisms for firmware and configuration updates. In addition, improved security concepts often result in poor usability which discourages users from relying on these concepts. Therefore, in this paper, we present an easy-to-use NFC-based configuration approach for IoT devices that is secured by appropriate security measures in software and hardware. Since industrial usage of such a configuration approach entails different requirements than home usage, we present and compare three different configuration processes. The applicability of our approach is demonstrated by two prototypical implementations, as well as a detailed security analysis. We also show that the imposed overhead due to the implemented security measures is negligible for most configuration updates.**

***Index Terms*—Near Field Communication, Internet of Things, Security, Configuration.**

I. Introduction

SECURITY aspects of the Internet of Things (IoT) and the lack thereof are a major issue due to the high number of potentially vulnerable devices. Although IoT devices are often resource constraint, they are still an enticing target for attackers since these devices are often used in botnets [1], [2]. In addition to that, each device in the IoT is equipped with some sort of sensor. This fact also increases the risk of attacks since adversaries may be interested in the provided sensor data, especially of Industrial IoT (IIoT) devices. Various studies show that between 10% and 40% of all scanned IoT devices are vulnerable to attacks because of issues such as using standard settings as well as username and passwords [3], [4] or due to exposing their configuration interface to the Internet [5]. Therefore, we consider the secured and easy-to-use configuration of IoT devices as a major gap in current research.

Regarding the configuration of IoT devices, we consider two application domains that entail different requirements in terms of security, hardware requirements, and usability.

(i) Industrial: Industrial usage of IoT devices requires high levels of security since malicious devices might interrupt a production process, reveal confidential information, or even cause physical damage and threaten human lives [6]. So-called smart factories [7] utilize a large number of IIoT devices for sensing the production process. Maintenance that involves configuration updates due to updated production- or security-requirements is essential in such an environment. By introducing a secured and easy-to-use configuration interface, even untrained staff can perform firmware updates or configuration changes. However, it is essential to protect the confidentiality and authenticity of configuration data as employees applying the configuration updates could be potential adversaries. Since in industrial settings the security aspect is of utmost importance, other factors such as the necessity for additional hardware components that increase the security can be seen as negligible.

(ii) Personal: Configuration approaches for IoT devices used in home automation or smart home [8] contexts need to provide good usability and low cost. However, also in a smart home context, configuration and firmware updates for devices need to be performed using a secured configuration interface. Similar to industrial use-cases, also in a smart home context the configuration data must be secured against various attacks for sustaining the proper functionality of the configured devices.

Independent of the domain in which IoT devices are used, configuration updates need to be performed in every phase of the device's lifecycle. Fig. 1 shows a typical IoT device configuration lifecycle that involves three major configuration phases: *initial configuration*, *reconfiguration*, and *deletion* of configuration data if an IoT device is sold or discarded. While the initial configuration might be performed in a controlled environment by the device manufacturer, all other reconfigurations of the IoT device will be performed in the potential presence of adversaries. Based on these observations, we extend and adopt the NFC-based configuration approach [9] presented at the IEEE International Conference on RFID. In addition to the configuration approach presented in that paper, we present different implementations that are tailored to the needs of certain application domains.

Contributions. For a configuration interface that is suitable for a wide range of IoT devices, we identify the following requirements:

T. Ulz, T. Pieber and C. Steger are with the Institute for Technical Informatics, Graz University of Technology, Graz, Austria. e-mail: {thomas.ulz, thomas.pieber, steger}@tugraz.at

S. Haas and A. Höller are with Infineon Technologies Austria AG, Graz, Austria. e-mail: {andrea.hoeller, sarah.haas}@infineon.com

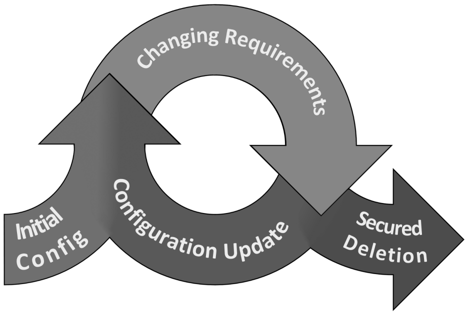

Fig. 1. Necessary configuration phases during an IoT device's lifecycle.

Req1 Configuration changes for IoT devices should be possible in a secured and easy-to-use manner.
Req2 The configuration interface must be protected against remote attacks and misuse.
Req3 Implemented security features should provide high usability such that users acceptance is improved.
Req4 Potential hardware extensions must be suitable for legacy devices as well as for newly developed devices.
Req5 Device configuration must be possible in every phase of an IoT device's lifecycle.
Req6 The configuration approach should be suitable for industrial as well as home usage.
Req7 There should be no or minimal additional hardware required to perform configuration updates.

In this paper, we present an NFC-based configuration approach is capable of fulfilling these seven requirements. The presented approach provides data confidentiality, integrity, and authenticity while being intuitive to use. To account for different application domains such as industrial usage or home usage, we present different implementations of our approach that optimizes usability, and security for the respective domain. The implemented security features comprise a secured configuration protocol as well as a hardware extension that includes tamper resistant hardware to further increase provided security. This hardware extension is applicable for legacy and new IoT devices and enables device configuration in every phase of an IoT device's lifecycle. To highlight the applicability of our configuration approach, we also present a novel smart factory inspired use case which we used as a demonstrator for our prototype.

Outline. The remainder of this paper is organized as follows: In Section II we give background information on methods and technologies included in our approach and discuss related work. Section III defines our system model and lists corresponding assumptions. We then present our NFC-based configuration approach in Section IV and compare three different realizations of that approach. The security features implemented in software and hardware are then presented in Section V. In Section VI we show a prototypical implementation of our approach that was also demonstrated using a smart factory environment. The evaluation of our approach that includes a security analysis is discussed in Section VII. Future work and a conclusion are given Section VIII.

II. Background and Related Work

A. Near Field Communication (NFC)

NFC is a contactless communication standard based on RFID technology that operates at a radio frequency of 13.56 MHz [10], [11]. The typical communication range of NFC is approximately 10 cm while supporting bit rates that are multiples of 106 kbps (up to 848 kbps). Although the communication range of NFC is limited, a range of approximately 10 m for active and 1 m for passive devices should be considered as a rule of thumb for possible eavesdropping [12]. In addition to eavesdropping, also other types of attacks such as man-in-the-middle, denial-of-service or replay attacks can be applied to unsecured NFC communication [13]. Despite these potential issues, NFC is used in various domains due to its intuitive device coupling mechanism that is easy to understand for humans [14]. The mobile payment sector [15] and mobile ticketing applications [16] are the most prominent applications of NFC; however, NFC is also seen as a future building block for the IoT to link the real world with the digital world [17].

B. Symmetric Cryptography

Symmetric Cryptography requires the same cryptographic key to be used for data encryption and decryption. Due to this, the used key is considered as shared secret between communicating parties and thus, needs to be kept private. The most widely used symmetric cryptographic algorithm is the Advanced Encryption Standard (AES) [18]. Algorithms for symmetric cryptography such as AES are capable of providing data confidentiality. In order to also provide data integrity and authenticity, symmetric cryptography needs to be combined with other security measures, such as Message Authentication Codes (MAC).

Authenticated Encryption (AE) combines symmetric cryptography with MACs in a secured way such that data integrity and authenticity can be provided in addition to data confidentiality [19]. AES provides specialized modes of operation such as AES-CCM or AES-GCM that are capable of providing AE.

C. Tamper Resistant Hardware

Cryptographic algorithms such as AES can be implemented efficiently in hardware with respect to performance, power consumption, and size requirements [25]. However, such hardware components might leak information that can be used to reveal used keys or other information [26]. In addition to these so-called side-channel attacks, also invasive physical attacks can be used to reveal confidential information [27]. Tamper resistant hardware [28] such as security controllers (SCs) can be used to provide protected execution environments as well as secured data storage that mitigate side-channel and physical attacks. However, since SCs are not as powerful as general purpose controllers or dedicated hardware components, splitting the execution environment into a secured world and a normal world is suggested [29]. This splitting principle by implementing SCs as external hardware modules that can then be combined with general purpose CPUs.

TABLE I
COMPARISON WITH RELATED WORK. THE CHARACTERISTICS REGARDING ARBITRARY PAYLOAD, PROVIDED SECURITY, SUITABILITY FOR PERSONAL USE, AND SUITABILITY FOR INDUSTRIAL USE ARE EVALUATED.

Related Work	Remarks	Arbitrary Payload	Provided Security	Personal Use	Industrial Use
[20], [21]	Device pairing information is exchanged; no security is provided.	✗	✗	✗	✗
[22]	Reprogramming CRFID firmware over the air. No security provided.	✗	✗	✗	✗
[23]	NFC peer-to-peer framework that allows arbitrary data to be transmitted.	✓	✗	✓	✗
[24]	Arbitrary configuration data possible; initial configuration not secured.	✓	✗ / ✓	✓	✗
[9]	Arbitrary configuration data possible; hardware and software security.	✓	✓	✓	✗ / ✓
This work.	Arbitrary configuration data possible; hardware and software security; multiple configuration mechanisms supported.	✓	✓	✓	✓

D. NFC-based Device Configuration

Although NFC is considered as an ideal technology for device pairing [12], using it for IoT device configuration is not that common. Most device pairing solutions (e.g. [20], [21]) have in common that only pairing information can be transmitted and that no security measures are integrated. Wu et al. [22] present an approach for reprogramming computational RFID (CRFID) tags over the air. The authors propose to use the Electronic Product Code (EPC) protocol to update the firmware of passive CRFID tags. The drawbacks of the presented approach that only complete firmware images can be flashed as well as missing security features. Serfass and Yoshigoe [23] present a framework for NFC communication in wireless sensor networks. This framework allows arbitrary data to be transferred using NFC but does not provide security measures. Haase et al. [24] propose to NFC-enabled mobile phones for NFC-based sensor and actuator configuration in smart home contexts. The authors also discuss security measures. However, the initial device configuration is unencrypted, and no key update mechanism is provided. Ulz et al. [9] present a QR and NFC-based hybrid configuration approach that implements security measures in hardware and software. This approach is further extended in this paper such that different update mechanisms are supported to suit personal and industrial usage scenarios. Also, an automated key derivation process is included. The discussed related work is compared with the approach presented in this paper in Table I.

III. SYSTEM MODEL AND ASSUMPTIONS

When designing a configuration interface for IoT devices, we are faced with the following three problems:

1) Configuration data for IoT devices might contain confidential information, especially when considering IIoT devices. This configuration data needs to be transferred using an untrusted channel including potential adversaries that eavesdrop or manipulate the transferred data.
2) IoT and IIoT devices might be operated in unsupervised environments, thus configuration data needs to be stored at the device such that confidential information cannot be extracted, even if adversaries have unlimited physical access to the device under attack.

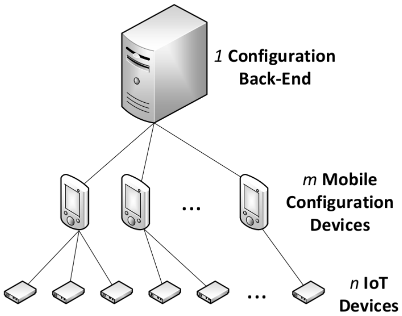

Fig. 2. System model we assume for IoT device configuration.

3) The configuration interface might be subjected to misuse, both unintentional and intentional.

To mitigate these problems, the configuration approach we are presenting in this paper is based on the system model shown in Fig.2 that comprises the following three entities:

IoT Device: The device that needs to be configured. There is no limitation on the number of devices; we generally assume n IoT devices in our system model.

Configuration Device: The mobile device used to transfer configuration data to the IoT device. We also do not limit the number of configuration devices in our model; therefore, we assume m such configuration devices.

Configuration Back-End: The back-end is responsible for administrating all configurations that are done using our presented approach. This means that the back-end needs to keep track of all configuration changes. Therefore, we assume *one* configuration back-end in our system model.

Based on our system model, we assume the configuration back-end that operates as a global configuration storage to be trustworthy and sufficiently secured against any kind of attack. We further assume that all configuration changes must be initiated and authorized by this back-end. Thus, the back-end has knowledge of device configurations from all devices administrated by that back-end. Regarding configuration data, no assumption concerning the content is made. That is, we assume configuration data to contain non-confidential informa-

```
temp_threshold: 0x5D
sampling_rate: 0x01
wifi_key: 0x778D9FE1325BAB9811
```

Fig. 3. Example configuration that contains non-confidential information as well as confidential information.

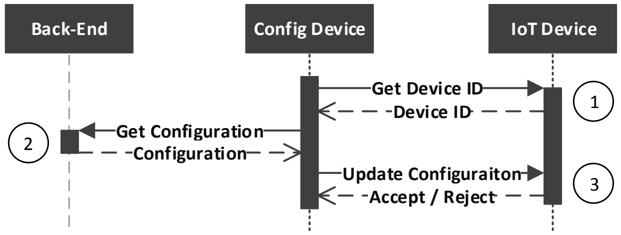

Fig. 4. Sequence of NFC communication for IoT device configuration.

tion such as temperature thresholds or sampling rates, as well as confidential information such as keys or WiFi passwords. An exemplary configuration is shown in Fig. 3.

IV. Configuration Mechanisms

Depending on the domain in which IoT devices are used, different requirements regarding a configuration interface can be defined. For example, protecting confidential information is of utmost importance for devices used in industrial settings. For IoT devices used by private persons, a configuration approach should be as easy-to-use as possible and should not require any costly additional hardware. Therefore, we present three different configuration mechanisms supported by our approach. Each of these mechanisms provides different advantages and disadvantages that we are going to discuss. All three mechanisms implement the security measures that are discussed in Section V. However, for simplicity, we only discuss the principle process of each configuration mechanism in this section.

A. NFC-based Configuration

In the NFC-based configuration mechanisms, NFC and a wireless data connection are used during device configuration according to the protocol shown in Fig. 4. The protocol comprises the following three steps: ① The configuration device queries the IoT device for its identifier using NFC. ② Using this device ID, a configuration is fetched from the configuration back-end. In this case, we assume that the device is managed by that back-end and that a new configuration that needs to be applied is available. ③ If a configuration for that given device is available; it is transferred to the IoT device again using NFC.

Advantages/Disadvantages

+ This mechanism is easy to use. The device to configure is automatically identified, and the corresponding configuration is fetched. Due to the fact that users only need to bring the mobile configuration device with a working data connection close to the IoT device, this approach is very well suited for remote support.
– Active data connection is required to fetch configuration data which might not be possible in industrial settings. Also, initial data such as a symmetric key needs to be synchronized between IoT device and configuration back-end (e.g. by manufacturer).

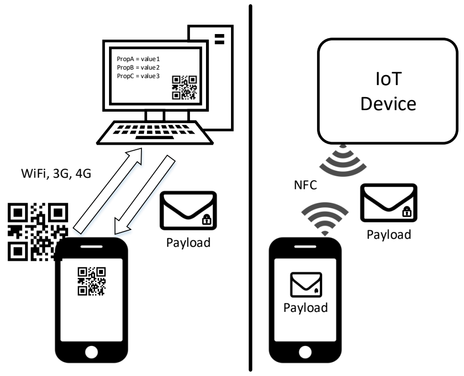

Fig. 5. Hybrid configuration approach: On the left hand side, configuration data is fetched from the back-end using a QR code. On the right hand side, the configuration is transfered to the IoT device using NFC.

B. NFC and QR code based Configuration

As third configuration mechanism, we propose a QR-code and NFC-based hybrid configuration approach [9]. The principle of that approach is shown in Fig. 5. As can be seen there, QR-codes are used to transfer configuration data from the configuration back-end to the mobile configuration device, while NFC is used to transfer the configuration data from the mobile configuration device to the IoT device. Due to the limited maximum payload of a QR code [30] and to support different usage scenarios, we propose the following two different modes of operation:

1) The complete configuration payload is stored in the QR code, which allows a maximum payload of roughly 2900 bytes of data. Therefore, we denote this type as *inline* QR code. Inline QR codes do not require the mobile configuration device to have an active network connection. Thus, these QR codes can be distributed and used where no working network is available.
2) If the configuration data is larger than the maximum payload of 2900 bytes, only an URL pointing to the configuration stored at the back-end is included in the QR code. The mobile configuration device then needs to fetch the configuration data from the back-end, as in the previous two mechanisms. We denote this type of QR code as *URL* QR code.

Advantages/Disadvantages

+ This mechanism is easy to use. In addition to that, if the inline mode is used, no active network connection is required when configuring devices. The inline mode allows using this approach in situations where no working network connection is available. QR codes can also easily be distributed by paper, for instance, by including the

TABLE II
COMPARISON OF PRESENTED CONFIGURATION MECHANISMS.

	NFC-based	Hybrid Inline	Hybrid URL	Location Aware
Device ID read	✓	✗	✗	✓
Works offline	✗	✓	✓/✗	✗
Configuration size unlimited	✓	✗	✓	✓
Suited for remote support	✓	✓	✓	✗
Complex back-end upkeep	✗	✗	✗	✓
Suited for personal use	✓	✓	✓	✗
Suited for industrial use	✓	✓	✓	✓
Camera required	✗	✓	✓	✗
Wireless interface required	✓	✗	✓	✓
Infrastructure required	✗	✗	✗	✓

initial configuration of a device inside the packaging the device is sold in. Also, as shown in Fig. 5, configurations can be directly downloaded from the monitor where a configuration is edited.

– The mobile configuration device needs to have a working camera in order to scan QR codes. Also, this mechanism potentially favors potential user errors since the user needs to be aware which configurations need to be downloaded beforehand when using the inline mode.

C. Location-Aware Configuration

The second implemented configuration mechanism does not require the device to identify itself. Instead, localization mechanisms are used to determine the mobile configuration device's position and the closest administrated IoT device. As soon as the configuration process is initiated by the user, the corresponding configuration is fetched from the configuration back-end as in the NFC-based approach. However, instead of requesting a configuration based on the IoT device's ID, the estimated coordinates of the mobile configuration device are sent to the configuration back-end. The back-end then replies with the most probable device configuration that is then sent to the IoT device using NFC.

Advantages/Disadvantages

+ This mechanism is easy to use. The device to configure is automatically identified, and the corresponding configuration is fetched. Due to using localization to identify the corresponding IoT device, NFC communication between IoT device and mobile communication device are reduced to a minimum.

– Active data connection is required to fetch configuration data which might not be possible in industrial settings. Also, the configuration back-end needs to be configured such that the location of each administrated device is known to the back-end. While outdoor localization using GPS might be accurate and easy, indoor localization is an ongoing research topic [31]. Also, indoor localization requires an infrastructure that is used to calculate the mobile configuration device's position.

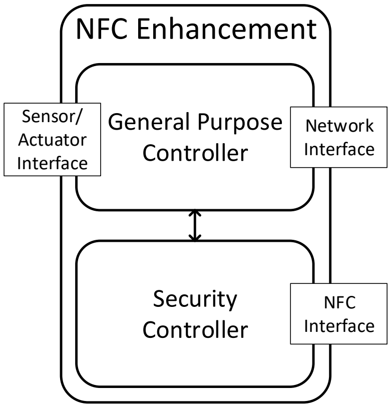

Fig. 6. NFC Enhancement that can be integrated into any IoT device.

In our prototypical implementation, we used a Received Signal Strength Indicator (RSSI) based trilateration algorithm [32] relying on wireless access points. The localization quality using such an approach strongly depends on factors such as fading, obstacles, or the temperature [33]. However, in our setting, we were able to achieve accuracies of less than 1 m which will be sufficient for most settings.

D. Comparison

Since all three previously presented mechanisms have different advantages and disadvantages, we compare them in Table II regarding their suitability for different usage scenarios of IoT devices. As can be seen there, no algorithm is suited best for all scenarios; thus, the applied mechanism needs to be chosen based on the context in which IoT devices need to be configured.

V. SECURITY MECHANISMS

A. NFC Enhancement

In order to allow new as well as retrofit IoT devices to be equipped with the proposed NFC configuration interface, we present a hardware extension suitable for these two types of devices. This so-called *NFC Enhancement* provides a number of interfaces for different purposes:

Sensor/Actuator Interface: This interface is used to connect sensors and actuators that are used by the IoT device with the NFC enhancement component.

Network Interface: This interface is used to connect the IoT device with a network. The IoT device's core functionality is accessible through this interface.

NFC Interface: This interface is used for device configuration. The NFC interface will also be used to harvest energy during the configuration process such that no additional power source is necessary for device configuration.

A concept of the NFC enhancement component containing all three interfaces is shown in Fig. 6. The component includes two controllers, a *general purpose controller* and a *SC*. Due to including two controllers, responsibilities can be split perfectly according to the capabilities of both controllers. On the one hand, the general purpose controller provides interfaces to sensors, actuators, and to the network which requires computational power. In addition, computational expensive data aggregation, manipulation, and processing can be done by the general purpose controller. The less powerful SC, on the other hand, offers a secured execution environment and protected storage. Cryptographic operations are executed by the SC, and confidential information is stored in the SC's protected storage. The SC also offers an NFC interface that is used for IoT device configuration. This NFC interface is also capable of harvesting energy from an NFC field such that the SC does not require any additional power source. Due to this, IoT devices can be configured at any time, for instance, during the manufacturing process without attaching any power source. In addition, SCs in our approach are considered as trusted entity, since their correct functionality is evaluated based on a common criteria [34] certification process. That is, all security critical operations performed by the SC in our approach can be considered as being properly secured and correct.

B. Data Transfer Protocol

Configuration data in our approach needs to be transferred using different untrusted channels (NFC, QR, WiFi, ...). Therefore, security measures need to be applied to provide data confidentiality, integrity, and authenticity. In addition to that, information regarding configuration data is necessary such that the IoT device's SC is capable of deciding if a configuration should be rejected or accepted and thus applied. For NFC data transfer we implement a protocol based on the NFC Data Exchange Format (NDEF) [35]. Due to the low overhead of NDEF, we use the same data structure when transferring data using QR-codes or a network connection. Although NDEF provides some security measures such as signatures [36], we did not rely on these measures since they are insufficient and shown to be vulnerable to attacks [37]. Instead, confidential information in our approach is protected by applying AE in the MAC-then-Encrypt mode that is also used in the Transport Layer Security (TLS) protocol [38]. The complete structure of NDEF packets in our approach including additional security related fields ins shown in Fig. 7. All but the **Cipher Spec** field is protected by AE in our approach. This field specifies the applied cryptographic algorithm and key size used for AE. This information needs to be transmitted unencrypted since it is required for decryption. All other fields are contained in the encrypted payload.

Version: A version number identifying the specific configuration version. The IoT device will reject configuration updates with a configuration number less or equal to the currently applied configuration.

Validity: If the current realtime of the IoT device is later in time than the specified validity, a configuration update will be rejected. For this check, we assume there is a secured time source for the IoT device.

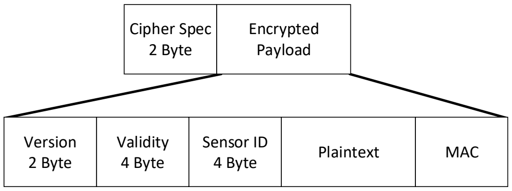

Fig. 7. NDEF packet structure used to protect configuration data.

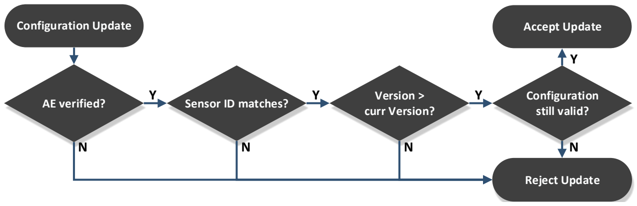

Fig. 8. Decision process of configuration update rejection or acceptance.

Device ID: If the specified device ID does not match the actual device's ID, the configuration is not indented for the respective device and thus rejected.

MAC: The MAC corresponding to the transmitted plaintext. It is calculated by a so-called one-way function [39] and is part of the AE process.

Using the additional information together with AE, the IoT device either rejects or accepts the configuration update. The flowchart in Fig. 8 summarizes the decision process.

VI. Prototype

To evaluate the presented approach with respect to feasibility, usability, and functionality we implemented a prototype comprising the presented security measures in hardware and software. This prototype, shown in Fig. 9 consists of the following components:

Sensor/Actuator: An air pressure sensor without any actuator is used to represent the IoT device's functionality.

General Purpose Controller: An Infineon XMC4500 microcontroller from the Cortex M4 family was used as a general purpose controller. This microcontroller provides connection interfaces such as USB, I^2C, and Ethernet.

SC: As SC, an Infineon SLE78 that is CC EAL5+ certified was used. The SC is connected to the general purpose controller via I^2C. The SLE78 SC provides security features such as secured data storage and code execution while also including a contactless interface for NFC communication.

Mobile Configuration Device: We used an off-the-shelf Nexus S mobile phone as NFC-enabled mobile configuration device. The device is running Android 4.1.2 Jelly Bean to use API level 14 and above that supports the latest NDEF functionality of Android.

Configuration Back-End: The configuration back-end that is not pictured in Fig. 9 was realized on a standard Windows PC in this prototype. The required functionality is written in NodeJS such that any computer that is capable of running JavaScript can run the back-end.

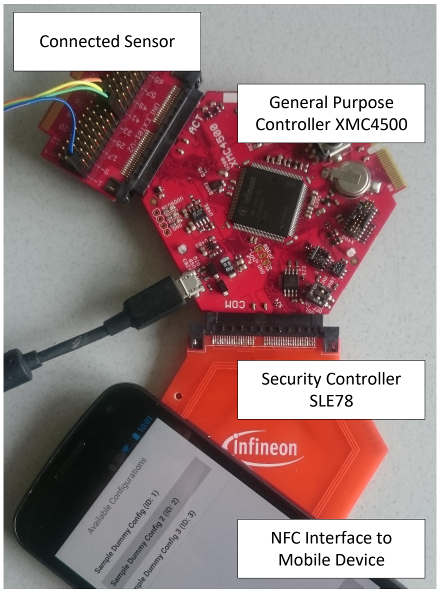
Connected Sensor
General Purpose Controller XMC4500
Security Controller SLE78
NFC Interface to Mobile Device
Infineon
Available Configurations
Sample Dummy Config (ID: 1)
Sample Dummy Config 2 (ID: 2)
Sample Dummy Config 3 (ID: 3)

Fig. 9. NFC Enhancement prototype comprising of an Infineon XMC4500 microcontroller used as general purpose controller and an Infineon SLE78 SC.

A. Smart Factory Prototype

In addition to the prototype shown in Fig. 9 we also present a prototypical smart factory use case in which we evaluated our presented approach. The evaluation is done in an *Industrie 4.0* [40] inspired smart factory setting that is simulated in the RoboCup Logistics League (RCLL) [41]. The league is intended as a testbed for smart factory inspired robotic solutions where a number of autonomous mobile robots need to transport semi-finished and finished individualized products between production machines. In this process, robots also need to configure machines such that the desired products are manufactured. The configuration in this context is done using wireless communication. The main issues with wireless communication in industrial settings are realtime capability and reliability [42]. Reliability of wireless communication is most often compromised by interference of various technologies operating in the same frequency spectrum [43]. Especially the 2.4 GHz and 5 GHz spectra are used by many technologies such as WiFi, Bluetooth, or wireless phones. Hence, when using these technologies interference is a common problem.

We applied the NFC-based configuration approach presented in this paper to the RCLL context and used an existing simulation environment [44] that we extended such that NFC related characteristics could be simulated [45]. Using this simulation environment, we investigated if the achieved production capacity of a system is negatively influenced if our NFC-based approach is used instead of WiFi. Fig. 10 shows the prototype during a simulation run. Since the point-to-point performance of our approach is comparable to WiFi (see Section VII), no drawback in terms of production capacity could be observed. However, due to the limited communication range of NFC, interference caused by robots simultaneously configuring machines was reduced, and thus, the reliability of machine configuration could be improved.

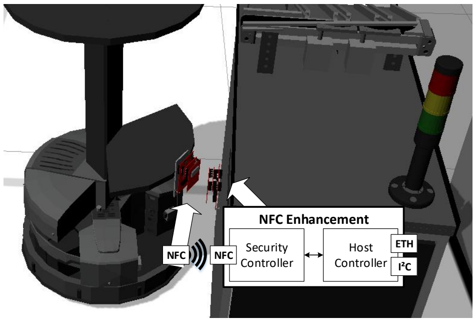

Fig. 10. Simulated smart factory prototype in RCLL environment.

VII. Evaluation

In addition to the presented prototype that demonstrates the feasibility and usability of our presented approach, we also evaluated the provided security level, the overhead, and the performance of our approach.

A. Security Analysis

To demonstrate the security improvements achieved by the implemented security measures in our approach, we present a comprehensive security analysis. In this analysis, we list involved **E**ntities, **A**ssets that are threatened and need to be protected, the **T**hreats, **C**ountermeasures applied to mitigate these threats, and **R**esidual Risks for threats that cannot be mitigated. We also list **As**sumptions that are made in the context of this analysis. An overview of the security analysis in Goal Structure Notation (GSN) is shown in Fig. 11. In this notation, the threats for each asset are highlighted. In addition, for each threat existing countermeasures or residual risks are shown. The assets that are protected by our secured configuration approach are: **(A1) Configuration Data:** Since configuration data may contain confidential information such as keys, the confidentiality, integrity, and authenticity of this information needs to be protected. **(A2) Device Functionality:** Correct functionality of IoT devices must not be compromised due to the inclusion of our proposed configuration interface. That is, any attack targeting this interface must not disturb proper operation of the device.

Threats for these assets can be posted by the following entities: **(E1) IoT Device Manufacturer:** The manufacturer of the device. Manufacturing includes all components such as sensors, actuators, and NFC enhancement. **(E2) IoT Device Owner:** Any user that is in possession of the IoT device and thus, allowed to make configuration changes. **(E3) Person Applying Configuration Updates:** Any person that is trying to apply configuration updates at the IoT device. This could be a different person than the device owner, especially in industrial settings. **(E4) Adversary:** Any adversary that can access the IoT device's configuration interface, either remotely or physically.

Before investigating potential threats, certain assumptions are made in order to restrict the scope of this threat analysis:

(As1) Configuration Back-End: The configuration back-end that maintains all current configurations is assumed to be properly secured against any type of attack. **(As2) SC Certification:** The SC used in the NFC Enhancement component is assumed to be certified to a CC security level of at least EAL5+ and thus, is capable of mitigating physical attacks.

Considering all identified assets, entities, and corresponding assumptions our approach is now analyzed regarding potential threats. For each threat, one or multiple countermeasures and potential residual risks will be given. **(T1) Backdoors:** There might be intentional or unintentional backdoors included in the configuration interfaces hardware or software. **(C1) CC Certification:** The CC certification process investigates and mitigates this type of threat. **(T2) Weak Cryptographic Algorithms:** The algorithms used in the configuration approach might be susceptible to attacks due to weaknesses in the used algorithm or due to using too short keys. **(C1) CC Certification:** The CC certification process investigates and mitigates this type of threat. **(T3) Bugs:** Security related functionality implemented by the manufacturer might include weaknesses or even bugs. **(C1) CC Certification:** The CC certification process investigates and mitigates this type of threat. **(T4) Security Breach:** Initial keys stored by the manufacturer could be lost in a security breach or disclosed in any other form, intentional or unintentional. **(C2) Easy Configuration:** Changing configuration parameters such as the initial key can be easily performed by users. **(R1) No Update:** If the initial key is not updated, this threat cannot be mitigated. **(T5) Eavesdropping Configuration Data:** An adversary might try to eavesdrop configuration data and thus, learn confidential information. **(C3) Security Measures:** The security measures presented in this publication provide effective mitigation of this threat. **(T6) Manipulate Configuration Data:** An adversary might try to manipulate transferred configuration data, either while being transferred from configuration back-end to mobile configuration device, or while being transferred from mobile configuration device to IoT device. **(C3) Security Measures:** The security measures presented in this publication provide effective mitigation of this threat. **(T7) Malicious Configuration:** An adversary might try to apply outdated configuration data or configuration data that is intended for a different device. **(C3) Security Measures:** The security measures presented in this publication provide effective mitigation of this threat. **(T8) No Update:** A malicious user does not apply any necessary update. Thus, he basically performs a denial-of-service (DoS) attack targeting the IoT device's correct functionality. **(R2) No Countermeasure:** Our approach cannot provide any countermeasure against users that do not apply intended updates. **(T9) DoS Attack:** An adversary might try to perform DoS attacks targeting the configuration interface. **(C4) Security Measures and Proximity:** The security measures presented in this publication provide effective mitigation of this threat. In addition, DoS attacks targeting the configuration interface can only be performed by adversaries in close proximity to the IoT device. **(T10) Physical Attacks:** An adversary that has physical access to the IoT device might try to reveal confidential information by performing physical attacks on the device. **(C1) CC Certification:** The CC certification process investigates and mitigates this type of threat.

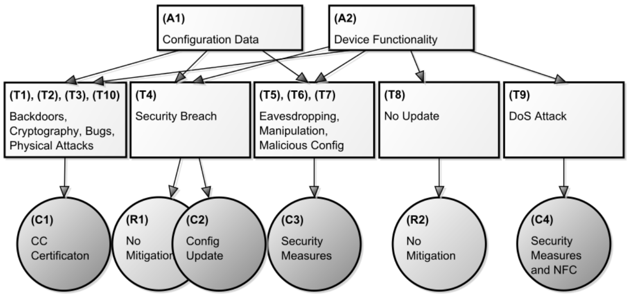

Fig. 11. Security analysis in GSN.

The list of discussed threats as well as the respective countermeasures and residual risks is not exhaustive by any means, but it reflects the threats that we consider as most crucial for the presented NFC-based configuration approach. Of the eleven identified threats, nine can be effectively mitigated while residual risks remain only for two threats. This highlights the improved level of security provided by the presented configuration approach.

B. Overhead and Performance

The implemented security measures in the NDEF protocol (see Section V-B) entail an overhead of transferred data. This overhead can be split into a static part (O_{static}) and a variable part ($O_{variable}$). The static overhead resulting from the additionally included information (cipher spec, version, validity, and sensor ID) can easily be calculated by summing up the field sizes specified in Fig. 7.

$$O_{static} = 2B + 2B + 4B + 4B = 12B \tag{1}$$

The variable overhead depends on the selected cryptographic algorithms and the corresponding key sizes. For this evaluation, we assume a MAC length of 32 B. In addition to that, also the padding required by block ciphers needs to be accounted for. For this evaluation, we assume AES that has a block size of 16 B which entails an overhead due to the padding of 0 B - 15 B. The total overhead O is then calculated by summing up all incidental overheads.

$$O = O_{static} + O_{dynamic} \tag{2}$$

An overview of the resulting overhead relative to the transferred configuration data size up to 4 kB is shown in Fig. 12. The sawtooth pattern results from the varying padding overhead that oscillates in the range of 0 B - 15 B. For typical configuration sizes of 300 B, less than 15 % of the transferred data will result from security imposed overhead.

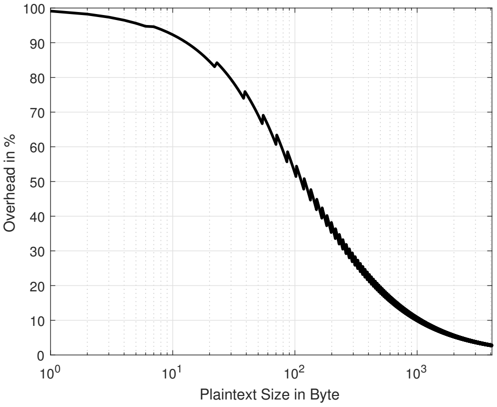

Fig. 12. Percentage of overhead relative to transferred configuration data.

To evaluate the performance of our approach, we measured the time that was required to transfer a configuration package with a typical size of 300 B. The complete data transfer including key agreement, encryption and decryption, and configuration acceptance/rejection decision process requires roughly 350 ms. Compared to that, transmitting the same amount of data using a secured TLS channel over a direct WiFi connection between two Raspberry PIs takes roughly 200 ms. However, it has to be considered that the processing power of a Raspberry PI is by far larger than the used components in our prototype and that a direct WiFi connection was used between the devices. Therefore, the timing difference between these two approaches can be assumed as negligible.

VIII. Future Work and Conclusion

In this paper, we present a secured NFC-based configuration approach that is suitable for personal and industrial IoT devices alike. To account for the different requirements in these two domains, we present different configuration mechanisms that provide different advantages and disadvantages. In order to provide data confidentiality, integrity, and authenticity we present security measures in hardware and software. The NFC enhancement component we present, can be used for new and retrofit IoT devices. The NDEF based protocol we present is secured by applying authenticated encryption in combination with additional information that is used to validate configuration data. The feasibility and usability of our approach are demonstrated by two prototypes, while the provided security, the resulting overhead, and the performance are also evaluated. As future work, we plan to further extend our approach such that the correct change of configuration data can be attested in our system.

Acknowledgment

This project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 692480. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Netherlands, Spain, Austria, Belgium, Slovakia.

References

[1] E. Bertino and N. Islam, "Botnets and Internet of Things Security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.

[2] D. Peraković, M. Periša, and I. Cvitić, "Analysis of the IoT Impact on Volume of DDoS Attacks," in *33rd Symposium on New Technologies in Postal and Telecommunication Traffic (PosTel 2015)*, 2015, pp. 295–304.

[3] A. Cui and S. J. Stolfo, "A Quantitative Analysis of the Insecurity of Embedded Network Devices: Results of a Wide-Area Scan," in *Proceedings of the 26th Annual Computer Security Applications Conference*. ACM, 2010, pp. 97–106.

[4] M. Patton, E. Gross, R. Chinn, S. Forbis, L. Walker, and H. Chen, "Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT)," in *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*. IEEE, 2014, pp. 232–235.

[5] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "IoTPOT: Analysing the Rise of IoT Compromises," *EMU*, vol. 9, p. 1, 2015.

[6] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and Privacy Challenges in Industrial Internet of Things," in *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*. IEEE, 2015, pp. 1–6.

[7] D. Lucke, C. Constantinescu, and E. Westkämper, "Smart Factory - A Step towards the Next Generation of Manufacturing," in *Manufacturing Systems and Technologies for the New Frontier*. Springer, 2008, pp. 115–118.

[8] R. Harper, *Inside the Smart Home*. Springer Science & Business Media, 2006.

[9] T. Ulz, T. Pieber, C. Steger, C. Lesjak, H. Bock, and R. Matischek, "SecureConfig: NFC and QR-Code based Hybrid Approach for Smart Sensor Configuration," in *RFID (RFID), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–6.

[10] I. O. for Standardization/International Electrotechnical Commission *et al.*, "ISO/IEC 18092 Information technologyâĂŤTelecommunications and information exchange between systemsâĂŤNear Field CommunicationâĂŤInterface and Protocol (NFCIP-1)," *ISO/IEC*, vol. 18092, 2004.

[11] ——, "ISO/IEC 21481 Information technologyâĂŤTelecommunications and information exchange between systemsâĂŤNear Field CommunicationâĂŤInterface and Protocol (NFCIP-2)," *ISO/IEC*, vol. 21481, 2005.

[12] E. Haselsteiner and K. Breitfuß, "Security in Near Field Communication (NFC)," in *Workshop on RFID security*, 2006, pp. 12–14.

[13] G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger, "NFC Devices: Security and Privacy," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*. IEEE, 2008, pp. 642–647.

[14] D. López-de Ipiña, J. I. Vazquez, and I. Jamardo, "Touch Computing: Simplifying Human to Environment Interaction through NFC Technology," *1as Jornadas Científicas sobre RFID*, vol. 21, 2007.

[15] J. Ondrus and Y. Pigneur, "An Assessment of NFC for Future Mobile Payment Systems," in *Management of Mobile Business, 2007. ICMB 2007. International Conference on the*. IEEE, 2007.

[16] V. Coskun, B. Ozdenizci, and K. Ok, "A Survey on Near Field Communication (NFC) Technology," *Wireless personal communications*, vol. 71, no. 3, pp. 2259–2294, 2013.

[17] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[18] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer Science & Business Media, 2013.

[19] M. Bellare and C. Namprempre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2000, pp. 531–545.

[20] A. Matos, D. Romao, and P. Trezentos, "Secure Hotspot Authentication through a Near Field Communication Side-Channel," in *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2012, pp. 807–814.

[21] M. Jung, J. G. Park, J. H. Kim, and J. Cho, "Interoperability between Medical Devices using Near Field Communication," in *2013 International Conference on Information Science and Applications (ICISA)*. IEEE, 2013, pp. 1–4.

[22] D. Wu, M. J. Hussain, S. Li, and L. Lu, "R2: Over-the-Air Reprogramming on Computational RFIDs," in *RFID (RFID), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–8.

[23] D. Serfass and K. Yoshigoe, "Wireless Sensor Networks Using Android Virtual Devices and Near Field Communication Peer-To-Peer Emulation," in *Southeastcon, 2012 Proceedings of IEEE*. IEEE, 2012, pp. 1–6.

[24] J. Haase, D. Meyer, M. Eckert, and B. Klauer, "Wireless sensor/actuator device configuration by NFC," in *2016 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2016, pp. 1336–1340.

[25] G. Rouvroy, F.-X. Standaert, J.-J. Quisquater, and J.-D. Legat, "Compact and Efficient Encryption/Decryption Module for FPGA Implementation of the AES Rijndael Very Well Suited for Small Embedded Applications," in *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, vol. 2. IEEE, 2004, pp. 583–587.

[26] S. Mangard, N. Pramstaller, and E. Oswald, "Successfully Attacking Masked AES Hardware Implementations," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2005, pp. 157–171.

[27] A. Becher, Z. Benenson, and M. Dornseif, "Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks," in *International Conference on Security in Pervasive Computing*. Springer, 2006, pp. 104–118.

[28] S. Ravi, A. Raghunathan, and S. Chakradhar, "Tamper Resistance Mechanisms for Secure Embedded Systems," in *VLSI Design, 2004. Proceedings. 17th International Conference on*. IEEE, 2004, pp. 605–611.

[29] M. Sabt, M. Achemlal, and A. Bouabdallah, "The Dual-Execution-Environment Approach: Analysis and Comparative Evaluation," in *IFIP International Information Security Conference*. Springer, 2015, pp. 557–570.

[30] T. J. Soon, "QR Code," *Synthesis Journal*, vol. 2008, pp. 59–78, 2008.

[31] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor Localization Without the Pain," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 2010, pp. 173–184.

[32] N. K. Sharma, "A Weighted Center of Mass Based Trilateration Approach for Locating Wireless Devices in Indoor Environment," in *Proceedings of the 4th ACM international workshop on Mobility management and wireless access*. ACM, 2006, pp. 112–115.

[33] N. Baccour, A. Koubâa, L. Mottola, M. A. Zúñiga, H. Youssef, C. A. Boano, and M. Alves, "Radio Link Quality Estimation in Wireless Sensor Networks: A Survey," *ACM Transactions on Sensor Networks (TOSN)*, vol. 8, no. 4, p. 34, 2012.

[34] D. Mellado, E. Fernández-Medina, and M. Piattini, "A common criteria based security requirements engineering process for the development of secure information systems," *Computer standards & interfaces*, vol. 29, no. 2, pp. 244–253, 2007.

[35] "NFC Data Exchange Format (NDEF)," NFC Forum, Tech. Rep. NDEF 1.0, 07 2006.

[36] M. Roland and J. Langer, "Digital Signature Records for the NFC Data Exchange Format," in *Near Field Communication (NFC), 2010 Second International Workshop on*. IEEE, 2010, pp. 71–76.

[37] C. Mulliner, "Vulnerability Analysis and Attacks on NFC-enabled Mobile Phones," in *Availability, Reliability and Security, 2009. ARES'09. International Conference on*. IEEE, 2009, pp. 695–700.

[38] H. Krawczyk, "The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?))," in *Annual International Cryptology Conference*. Springer, 2001, pp. 310–331.

[39] M. Naor and M. Yung, "Universal One-Way Functions and their Cryptographic Applications," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*. ACM, 1989, pp. 33–43.

[40] T. Bauernhansl, M. Ten Hompel, and B. Vogel-Heuser, *Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung· Technologien· Migration*. Springer-Verlag, 2014.

[41] T. Niemueller, D. Ewert, S. Reuter, A. Ferrein, S. Jeschke, and G. Lakemeyer, "RoboCup Logistics League Sponsored by Festo: A Competitive Factory Automation Testbed," in *RoboCup 2013: Robot World Cup XVII*. Springer, 2014, vol. 8371, pp. 336–347.

[42] A. Frotzscher, U. Wetzker, M. Bauer, M. Rentschler, M. Beyer, S. Elspass, and H. Klessig, "Requirements and current solutions of wireless communication in industrial automation," in *Communications Workshops (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 67–72.

[43] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan, "Understanding and Mitigating the Impact of RF Interference on 802.11 Networks," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 385–396, 2007.

[44] F. Zwilling, T. Niemueller, and G. Lakemeyer, "Simulation for the RoboCup Logistics League with Real-World Environment Agency and Multi-level Abstraction," in *Robot Soccer World Cup*. Springer, 2014, pp. 220–232.

[45] T. Pieber, T. Ulz, and C. Steger, "SystemC Test Case Generation with the Gazebo Simulator," in *Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH), 2017 7th International Conference on*. IEEE, 2017, in Press.

Thomas Ulz received his master's degrees (M.Sc.) in Information and Computer Engineering as well as in Computer Science from Graz University of Technology, both in 2016. The focus of his studies included Security, Embedded Systems, Robotics, and Machine Learning. Currently, he is a Ph.D. student in Information and Computer Engineering at the Institute for Technical Informatics, Graz University of Technology. His research currently focuses on security aspects of IoT devices.

Thomas Pieber received his master's degree (M.Sc.) in Information and Computer Engineering from Graz University of Technology in 2016. The focus of his studies included Embedded Systems, Mobile Computing, and Robotics. Currently he is a Ph.D. student in Information and Computer Engineering for the Institute for Technical Informatics, Graz University of Technology. His research currently focuses on energy efficiency of IoT devices.

Andrea Höller received her master's degree (M.Sc.) in Information and Computer Engineering from Graz University of Technology, focusing on System-on-Chip Design and Information Security in the year 2013. From 2013 to 2016 she has conducted research on dependability for cyber-physical systems at the Institute for Technical Informatics. In 2016, she earned her Ph.D. degree at Graz University of Technology with a thesis on software-based fault-tolerance for resilient embedded systems. In September 2016, she joined Infineon Technologies Austria AG where she currently is working on the future of secured authentication and encryption for cyber-physical systems and the Internet of Things.

Sarah Haas received her master's degrees (M.Sc.) in Information and Computer Engineering as well as in Computer Science from Graz University of Technology, both in 2016. The focus of her studies included Robotics, Machine Learning, Embedded Systems and Big Data Analysis. Currently, she is a Ph.D. student in Information and Computer Engineering, working at Infineon Technologies Austria AG. Her research currently focuses on security of industrial mobile robots.

Christian Steger received the Dipl.-Ing. degree (M.Sc.) in 1990, and the Dr. techn. degree (Ph.D.) in electrical engineering from Graz University of Technology, Austria, in 1995, respectively. He graduated from the Export, International Management and Marketing course at Karl-Franzens-University of Graz in June 1993 and completed the Entrepreneurship Development Program at MIT Sloan School of Management in Boston in 2010. He is strategy board member of the Virtual Vehicle Competence Center (ViF, COMET K2) in Graz, Austria. Since 1992 he has been Assistant Professor at the Institute for Technical Informatics, Graz University of Technology were he heads the HW/SW codesign group at the Institute for Technical Informatics.