Hardware-Secured Configuration and Two-Layer Attestation Architecture for Smart Sensors

Thomas Ulz, Thomas Pieber, Christian Steger
Institute for Technical Informatics
Graz University of Technology
Graz, Austria
{thomas.ulz, thomas.pieber, steger}@tugraz.at

Sarah Haas, Rainer Matischek, Holger Bock
Development Center Graz
Infineon Technologies Austria AG
Graz, Austria
{sarah.haas, rainer.matischek, holger.bock}@infineon.com

***Abstract*—The necessity to (re-)configure Internet of Things devices such as smart sensors during their entire lifecycle is becoming more important due to recent attacks targeting these devices. Allowing configuration parameters to be changed in any phase of a smart sensor's lifecycle allows security updates or new key material to be applied. Also, the functionality of a smart sensor can be altered by changing its configuration. The challenges that need to be considered when enabling the configuration of arbitrary parameters are the security and usability of the configuration interface, the secured storage of confidential configuration data, and the attestation of successfully applied configuration updates. Therefore, we present an NFC-based configuration approach that relies on dedicated secured hardware to solve these challenges. In addition to a hardware extension for smart sensors, we also present a secured configuration protocol as well as a two-layer configuration attestation process to verify the correct utilization of all transmitted configuration parameters.**

***Index Terms*—Smart Sensor; Configuration; Attestation; Hardware Security.**

I. Introduction

Sensors are seen as one of the major building blocks of the Internet of Things (IoT) [1] where devices can be used to interact with their physical environment only due to embedded sensors. These sensor-equipped devices have led to technologies such as wireless sensor networks (WSN) and high-tech strategies such as Industry 4.0. In these technologies and high-tech strategies, sensing nodes often perform some sort of (pre-)processing in order to optimize properties such as their exactness, energy efficiency, or usefulness. Such sensors are also denoted as *smart sensors* [2].

Due to recent attacks targeting these devices, frequent reconfiguration is needed to mitigate certain kinds of attacks [3], [4]. For example, frequent changes of applied encryption keys or parameters such as a used elliptic curve could make attacks harder. Also, security related software updates will be needed to account for new security requirements. In addition to security related updates, also updated functionality of devices can be achieved. Rapidly changing environments, as well as frequently updated requirements regarding their operation, require smart sensors to be configurable. Weyer et al. [5] state that configuring devices will be essential for future Industry 4.0 motivated production systems.

One way to achieve the goal of flexible smart sensors is to make them self-configuring and adaptive [6]. Lee et al. [7] suggest self-configuration and self-adjustment as one of five major building blocks for cyber-physical systems (CPS) used in Industry 4.0 scenarios. However, self-configuration of smart sensors is not considered as mature enough to account for all requirements of industrial scenarios where higher safety and security standards need to be fulfilled. Therefore, manual configuration mechanisms that are reliable while providing a secured update process will be needed for smart sensors.

The European research project *IoSense*[1] addresses the configurability of smart sensors. As envisioned in the IoSense project, the configuration of smart sensors should be possible throughout the complete lifecycle of a sensor. The four phases of a smart sensor's lifecycle and potential example use-cases where a (re)configuration is needed are shown in Fig. 1.

To allow smart sensors to be configured during all four shown lifecycle phases, we propose a Near Field Communication (NFC)-based configuration approach that uses a dedicated hardware-based secure element to provide a protected execution environment for involved security critical code as well as secured storage for confidential configuration data. For non-confidential configuration data, storage will be provided by a general purpose computing environment. Due to these different storage layers and to provide efficient configuration attestation with minimal communication overhead, we also propose a two-layer configuration attestation architecture. Summarized, the contributions of this paper are:

- We present an NFC-based configuration approach suitable for smart sensors used in industrial environments. To account for the enhanced security requirements of industrial scenarios, a hardware architecture using dedicated hardware-based secure elements in combination with suitable cryptographic methods are used to provide data confidentiality, integrity, and authenticity.
- Due to providing unsecured as well as secured storage for configuration data, and to impose an overhead as small as possible, a two-layer configuration attestation architecture is presented in this paper. The configuration protocol is *attestation aware* to support the configuration attestation.

[1]http://www.iosense.eu

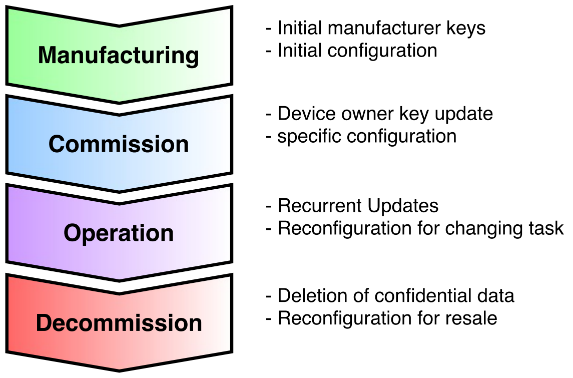

Manufacturing
- Initial manufacturer keys
- Initial configuration
Commission
- Device owner key update
- specific configuration
Operation
- Recurrent Updates
- Reconfiguration for changing task
Decommission
- Deletion of confidential data
- Reconfiguration for resale

Fig. 1. Four phases of a sensor's lifecycle with configuration scenarios.

TABLE I
COMPARISON WITH DEVICE CONFIGURATION RELATED WORK.

Related Work	Energy Efficient	Arbitrary Payload	Secured Protocol	Tamper Resistant	Attes-tation
[8], [9]	✗	✓	✓	✗	✗
[10]	✗	✓	✓	✗	✗
[11]	✓	✗	✗	✗	✗
[12]	✓	✓	✗	✗	✗
[13]	✓	✓	✓	✗	✗
[14]	✓	✗	✓	✓	✗
This work	✓	✓	✓	✓	✓

To the best knowledge of the authors, neither an attestation aware configuration approach nor a two-layer configuration attestation architecture were proposed in other works. The remainder of this paper is structured as follows. In Section II, background on involved technologies as well as related work is discussed. The general configuration and attestation problem, as well as a system model and corresponding assumptions, are discussed in Section III. Section IV demonstrates our proposed hardware-secured configuration approach, Section V presents the corresponding two-layer configuration attestation architecture. The presented approach is then evaluated in Section VI by means of a demonstrator. This paper is then concluded with Section VII.

II. Background and Related Work

A. Device Configuration

Device configuration is an important topic in the IoT due to the large number of devices. Many solutions have been proposed (e.g. [8], [9]) that define interfaces such that devices can be easily configured via the Internet. However, as most of these solutions expose the configuration interface of devices to the Internet without considering protocol *and* physical device security, they are not suitable for industrial scenarios. Steger et al. [10] propose a software update approach for vehicles that relies on 802.11s mesh networks that allow parallel updates of multiple cars. However, this approach is not suitable for industrial smart sensor configuration due to the following two reasons: (i) Due to many sensors being resource constraint devices that operate on battery power, 802.11 based technologies are considered to be too energy consuming for smart sensor configuration purposes. (ii) The authors consider adding hardware security modules (HSM) in their approach but state that such a solution would lead to significant extra costs. This is, of course, infeasible for smart sensors.

Configuration scenarios involving resource constraint devices often use NFC due to the fact that NFC devices operated in a passive mode provide excellent energy efficiency. Wu et al. [11] present an approach to reprogramming computational RFIDs over the air using the electronic product code protocol. Serfass and Yoshigoe [12] propose an Android-based framework for NFC peer-to-peer communication that allows transferring arbitrary data. Similar to that, Haase et al. [13] present an NFC-based configuration framework for sensors and actuators used in home automation contexts. However, due to the home automation focus, the security level provided by that approach is considered as insufficient for industrial scenarios. Ulz et al. [14] present a key update process for industrial devices based on NFC. However, this approach does not allow arbitrary configuration data to be transferred.

None of the presented approaches includes a verification process to ensure the correct application of new configuration data. An overview summarizing the related work regarding device configuration is shown in Table I.

B. Configuration Attestation

The remote attestation of device characteristics such as hardware properties, operating system, or services is a well-covered topic in the IoT [15], for wireless sensor networks [16], and generally for resource constraint devices [17]. Saroiu and Wolman [18] discuss various scenarios that are affected by untrustworthy sensor data. The authors also suggest to include trusted computing hardware such as a trusted platform module (TPM), Intel's trusted execution technology (TXT), or ARM's TrustZone (TXT and TrustZone either use or closely relate to TPM functionality [19]) into sensors to provide trusted data.

In fact, most proposed attestation approaches rely on trusted computing hardware, due to the constraints and assumptions that are often necessary for software-based attestation [20]. However, many approaches attest static parts of a system, such as the BIOS, boot loader, or binaries that should get executed, while we need to attest a configuration that is changing.

Regarding the attestation of changing properties, Kil et al. [21] propose a method for dynamic system properties attestation. In their approach, a *challenger* requests an attestation that is then performed by the *attester*. The dynamic properties that are attested are structures in an application's memory that need to be defined before deployment of the application. The method also needs a BIOS that supports *core root of trust measurements* which makes it infeasible for smart sensors. SCUBA, a secure code update by attestation for sensor networks [22] relies on indisputable code execution which is a software security measure shown to be susceptible to certain attacks [20]. A promising approach is so-called *property-based* attestation [23] that however requires functionality not yet included into the TPM specification. As an alternative, the authors assume a trusted execution environment that is needed in their approach, which we will include in our proposed architecture.

C. Near Field Communication (NFC)

NFC is a contactless communication standard based on several RFID standards. The technology is well-known for its usage in contactless payment solutions, ticketing, and access control systems [24]. NFC operates at 13.56 MHz, typically in ranges of 3-10 cm while supporting bit rates that are multiples of 106 kbps (up to 848 kbps). Due to the frequency spectrum used, NFC is not susceptible to interference from other wireless technologies such as WiFi, Bluetooth, or 801.15.4 based protocols. Due to the limited communication range, NFC provides certain security advantages compared to other wireless technologies [25]. Although the communication range of NFC is limited, attacks that allow eavesdropping in distances of 10 m haven been shown. Therefore, security measures to protect the confidentiality, integrity, and authenticity of transferred data need to be implemented.

D. Authenticated Encryption (AE)

AE comprises *private key cryptography* with *Message Authentication Codes (MAC)* in a secured way such that the confidentiality, integrity, and authenticity of data can be provided [26]. The well-known private key cryptography algorithm *Advanced Encryption Standard (AES)* provides specialized modes of operation such as AES-CCM that are capable of providing AE. AES can be implemented efficiently in hardware with respect to performance as well as size requirements (e.g. Rogawski and Gaj [27]). Therefore, the usage of dedicated hardware to perform security relevant operations in smart sensors is highly practicable [28].

E. Secure Element (SE)

The combination of processing units for secured code execution and secured storage for data and applications is denoted as SE. In contrast to general purpose CPUs, the secured code execution environment mitigates exploits based on flaws such as buffer overflows. In addition to that, the SE also implements appropriate countermeasures to mitigate physical attacks. SEs that are capable of mitigating physical attacks provide so-called *tamper resistance* [29]. The security level provided by SEs is assessed by a common criteria (CC) information technology security evaluation [30] in order to be able to compare the security provided by SEs.

III. Problem Definition and System Model

Before presenting our approach for smart sensor configuration and configuration attestation, we define the problem we face, and define our system model.

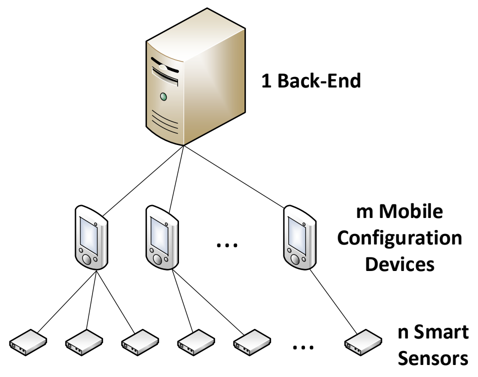


Fig. 2. System model for smart sensor configuration.

A. Problem Definition

When configuring smart sensors, (confidential) configuration data is transferred to a device that can not be fully trusted, even when trusted hardware components such as a trusted platform module (TPM) are included in the smart sensor. The configuration data also needs to be transferred to the device using a communication channel with potential adversaries. Therefore, we need to consider the following three problems:

1) To configure a smart sensor, confidential configuration data needs to be sent using a communication channel that might be accessible to potential adversaries.
2) A malicious sensor device might read configuration data and reveals confidential information to a third party.
3) A malicious sensor device might accept a configuration but does not apply it. Therefore, the correct functionality of the device is compromised.

To summarize these problems, we assume an adversary that is able to access and perform malicious operations on both the communication channel, and the smart sensor.

B. System Model

For our configuration approach we assume the system model shown in Fig.2 that comprises the following three entities:

Smart Sensor: The smart sensor that needs to be configured. There is no limitation on the number of devices; we generally assume *n* smart sensors in our system model.

Mobile Configuration Device: The mobile device used to update configuration data on smart sensors. In our approach there is no limitation regarding the number of configuration devices used, so we assume a number of *m* mobile configuration devices in our system model.

Configuration Back-End: The back-end manages and initializes all configuration changes. This means, changes need to be done done on the back-end from where they are transferred to the smart sensor using the mobile configuration device. In our system model, we assume *one* configuration back-end.

C. Assumptions

Based on our system model, we assume a back-end that operates as a global configuration storage to be trustworthy

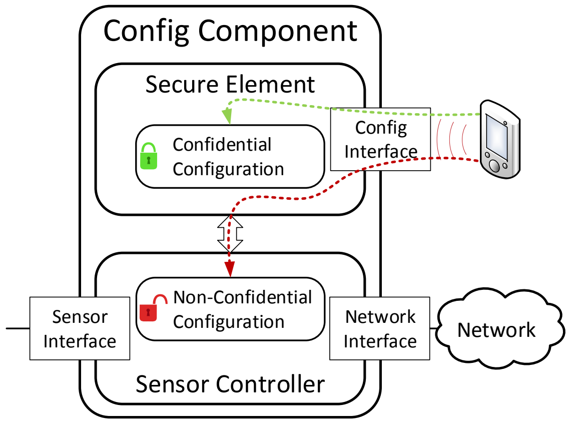
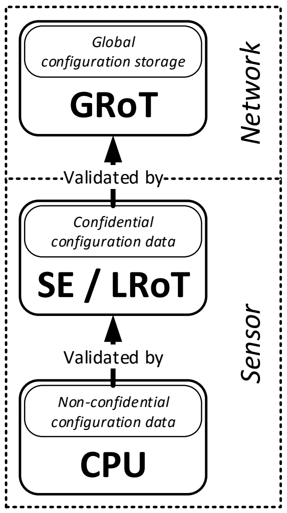


Fig. 3. NFC-based configuration architecture for a smart sensor's confidential as well as non-confidential configuration data.


Global configuration storage
GRoT
Network
Validated by
Confidential configuration data
SE / LRoT
Validated by
Non-confidential configuration data
CPU
Sensor
Fig. 4. Necessary two-layer architecture for configuration attestation due to allowing data of two confidentiality levels.

and sufficiently secured against attacks. We further assume that all configuration changes must be initialized and therefore authorized by this back-end. Thus, the back-end has knowledge of all smart sensor configuration versions that are configured and managed by the given back-end.

IV. NFC-BASED CONFIGURATION

Smart sensors need to be configured during their entire lifecycle as shown in Fig. 1. To account for this requirement, we propose to use an NFC-based configuration interface for the following reasons:

- NFC allows ad-hoc connections to be established instead of exposing the configuration interface to potential adversaries located in a network.
- The limited communication range of NFC also offers advantages in limiting the malicious use of this interface due to adversaries having to be in close proximity to the smart sensor in order to use to the configuration interface.
- If the NFC module of the smart sensor is operated in passive mode, no energy is needed for communication. The hardware components involved in the configuration process can even be powered through the NFC field of the communication partner's NFC device, which is needed in certain phases of a smart sensor's lifecycle (e.g. during manufacturing of the smart sensor).

A potential drawback of NFC is that there are no security measures included in the NFC standard to protect the confidentiality, integrity, and authenticity of transferred data. Therefore, we propose to use AE in combination with ticketing information that is used to verify if a configuration should be accepted by a smart sensor or not. To perform all involved cryptographic methods in a secured execution environment, our approach relies on an SE that is combined with a general purpose processor as shown in Fig. 3. This *Config Component* implements *security by isolation* approach (e.g. Vasudevan et al. [31]; *normal* and *secured* world) allows execution and data storage to be split into confidential or critical, and non-confidential or non-critical parts. The responsibility of the SE and sensor controller in our presented approach are:

SE: The SE offers a secured execution environment for critical code such as AE. In addition, the SE also offers secured storage for confidential configuration data that can be stored in a tamper resistant manner. To enable configuration transfer via NFC, the SE also includes an NFC interface. That interface allows the SE to be powered by an NFC field, even if there is no power source attached to the smart sensor. Due to the SE providing the NFC interface, confidential data is directly transferred to the SE and no additional interface for configuration updates or storage needs to be exposed which potentially also mitigates so-called API-level attacks that target these interfaces [32].

Sensor Controller: The sensor controller includes interfaces to the sensor hardware, to the network, and to the SE. Due to the fact that this controller is a general purpose controller, it also provides an execution environment for non-critical code as well as storage for non-confidential configuration data.

Having both, secured and unsecured data storage, our approach is able to handle confidential as well as non-confidential configuration data. Confidential configuration data could include information such as keys used for communication, firmware updates for the SE, or data for local decision making in a smart sensor. Non-confidential data could, for instance, represent settings such as the sampling rate of a smart sensor but also firmware updates for the sensor controller. Due to having two layers of configuration data with different confidentiality requirements, the verification process of the applied configuration update also needs to be done in a two-layer architecture as shown in Fig. 4. There, *LRoT* denotes the *local root of trust* that is used to attest non-confidential configuration data. *GRoT* denotes the *global root of trust* that is then used for confidential configuration attestation.

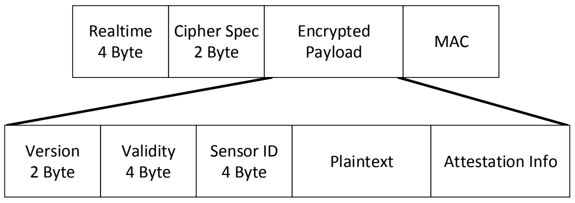


Fig. 5. NDEF packet structure of transferred configuration data.

Independent of the confidentiality level when storing configuration data, security measures need to be implemented when transferring the configuration data using NFC to mitigate eavesdropping, and replay attacks. Therefore, every configuration that is transferred needs to be secured using the security mechanisms shown in the NFC data exchange format (NDEF) packet structure (see Fig.5) used in our approach. The fields included in this NDEF packet are:

- *Realtime:* The realtime of the device that is sending the configuration update. This information is used to decide if a packet will be accepted or rejected by a smart sensor (see field *Validity*). This information is not encrypted as it is added to the NDEF package by the device deploying the configuration due to most smart sensors not having time synchronization.
- *Cipher Specifications:* This field defines the used encryption algorithm and corresponding key lengths to allow the smart sensor to use the appropriate algorithms when decrypting and verifying the package.
- *Encrypted payload and MAC:* These two fields are generated by encrypting the plaintext payload and by calculating the MAC of that same payload.
- *Version:* To mitigate replay attacks, a version number is included in the encrypted payload. For a smart sensor to accept a configuration update, this version number must be larger than the current configuration version.
- *Validity:* The validity defines how long a given configuration package is valid to also mitigate replay attacks. The validity is checked against the included realtime from the configuration deploying device.
- *Sensor ID:* The included sensor ID must match the sensor ID stored in the tamper resistant memory of the SE for the smart sensor to accept the configuration update.
- *Plaintext:* This is the transferred configuration data.
- *Attestation Info:* To allow the LRoT to perform attestation operations, attestation information is also added to the transferred data. The used attestation approaches will be discussed in detail in the next section.

The secured NDEF data structure is used to transfer data from the back-end to the mobile device as well as when transferring data from the mobile configuration device to the smart sensor. Using the same package entails that the transferred data can not be modified in any way on the mobile device. Since we consider the mobile configuration device itself as well as its operator as untrustworthy, only allowing data to be transferred secured by AE mitigates attacks enabled by malicious devices or users.

V. Two-Layer Configuration Attestation

Before presenting our two-layer configuration attestation architecture, we are briefly going to discuss some terminology related to attestation.

A. Attestation Terminology

Usually in attestation there are two roles, a *challenger* and an *attestor* [33]. The challenger is the entity interested in the correctness or trustworthiness of a system. That is, the output of the attestation process. The attestor (often also prover) is the entity that needs to prove its correctness and trustworthiness by measuring and attesting its configuration. An attestation process usually is assisted by some dedicated hardware that supports *trusted computing*. The TPM specification of the trusted computing group (TCG) lists two mechanisms that are of interest when discussing device attestation: *remote attestation* and *sealed storage*. Remote attestation defines how to use a TPM's secured storage, the platform configuration registers (PCR), to implement an attestation process. Sealed storage refers to data (information or code) that is stored encrypted using a key calculated as a function of a TPM's PCR values. That is, the data is only unsealed if the attestor is able to prove its correct state.

B. SE versus TPM

Most attestation solutions require secured hardware to be used at the attestor. This secured hardware component is a TPM in most cases. Sadeghi et al. [34] argue that such secured hardware is too complex and often too expensive for most resource constraint devices such as smart sensors. The authors also state that although software-based attestation solutions have been proposed, at least a basic subset of security features in hardware will be needed. Therefore, we propose to use an SE such as a product from Infineon's SLE78 product family (see [35]) in our approach for the following three reasons:

1) When using an SE such as Infineon's SLE78 that was designed for smart cards, secured hardware can also be included into smart sensors that are constrained in terms of size and available energy.
2) Although TPMs with NFC capability have been proposed [36], no currently available TPM offers an NFC interface. In contrast to that, certain SEs such as from the SLE 78 family offer an NFC interface and the required security properties needed for attestation.
3) The applied attestation approach that will be presented in this paper requires a trusted execution environment which is not included in the TPM specifications. However, security controllers are capable of providing such a tamper resistant execution environment.

C. Two-layer Approach

Our two-layer configuration attestation approach is based on the fact that the configuration solution presented in Section IV supports two different levels of confidentiality for configuration data. As shown in Fig. 4, non-confidential configuration will be attested using the SE included in our proposed configuration component while confidential configuration data will be attested by the trusted back-end.

D. *Non-Confidential Configuration Attestation*

Non-confidential configuration data that can be changed on the sensor controller can be either application updates (binaries), configuration data (e.g. sampling rate), or both. Since malicious code, as well as malicious configurations, can be harmful, attestation is necessary for both components. Due to the different nature of information, we suggest using two different attestation techniques respectively.

Application (Binaries) are static memory content that is changed less frequent than configuration data. According to Yang et al. [37] the size needed for application binaries can usually be assumed to be two orders of magnitudes larger than the space required for (configuration) data. Therefore, we propose to use basic binary attestation where a hash value is computed over the complete application binaries. Advanced methods such as *pseudo-random memory traversal* [38] can also be implemented to prevent attacks such as memory copy attacks, or pre-computation and replay attacks. The necessary information to attest the correctness of updated binaries are included in the configuration update NDEF package (attestation info, see Fig. 5) and therefore are also updated in the SE (LRoT) whenever new application binaries or non-confidential configuration data are transferred to the SE.

Non-Confidential Configuration Data is also stored in the sensor controller's memory. However, since configuration data is much smaller and attacks such as memory copy attacks are easy to implement, we propose to use *property-based attestation* [23] for configuration data. As stated by the authors, a property-based attestation mechanism requires additional functionality that is currently not included in the TCG's TPM specifications. Therefore, a trusted execution environment is needed to implement the desired functionality. In our approach, the trusted execution environment is provided by the SE. To implement property-based attestation, certificates are needed for each valid configuration property. The problem of certificate revocation can easily be solved in our approach by including certificate information in the configuration update's attestation information field.

The two different attestation techniques are then jointly used to *grant or deny network access* to the sensor controller. We suggest achieving this by using *sealed storage* to protect code such as the whole network stack, or other information from being accessed by an unattested sensor controller. By restricting network access through local attestation instead of using remote attestation, malicious smart sensors can be isolated from the network. Thus, such sensors are hindered from infecting other network devices or start network-based attacks such as denial-of-service attacks, jamming, or deception attacks [39], [40]. The decision on which information to seal in order to protect network access needs to be based on a trade-off between parameters such as security level, and overhead. On the one hand, sealing the network stack would require a sensor controller to prove its correctness only once before copying the network stack to its own memory. However, an adversary then could modify binaries or configuration after having unsealed the network stack. On the other hand, sealing an encryption key and requiring an attestation every time before allowing an encrypted packet to be sent, increases the overhead while still allowing certain kinds of attacks such as DoS attacks. Since we only present an attestation architecture in this paper, we refer to future work for a detailed comparison of different approaches. Independent of the chosen sealing approach, attestation information is stored in an SE that provides tamper resistance, attestation information is efficiently protected from being tampered with. Therefore, adversaries are not able to manipulate stored attestation information.

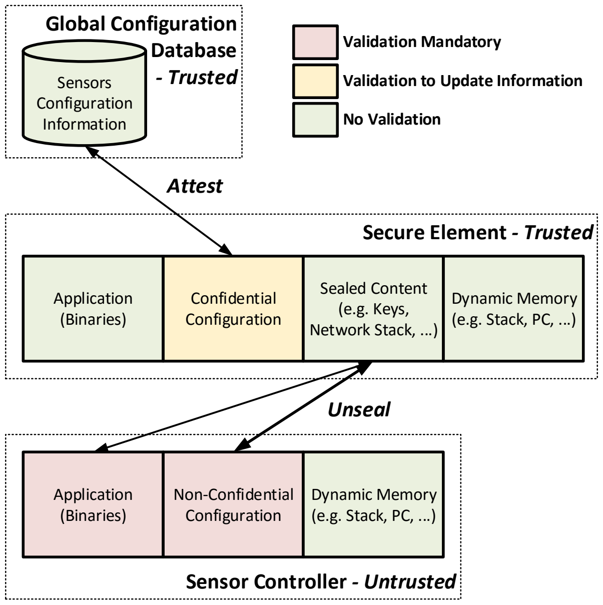

Global Configuration Database - Trusted
Sensors Configuration Information
Validation Mandatory
Validation to Update Information
No Validation
Attest
Secure Element - Trusted
Application (Binaries)
Confidential Configuration
Sealed Content (e.g. Keys, Network Stack, ...)
Dynamic Memory (e.g. Stack, PC, ...)
Unseal
Application (Binaries)
Non-Confidential Configuration
Dynamic Memory (e.g. Stack, PC, ...)
Sensor Controller - Untrusted

Fig. 6. Two-layer configuration attestation in detail.

E. *Confidential Configuration Attestation*

Confidential configuration data is secured by the applied security measures when transferring the data via NFC and storing that data on an SE that provides tamper resistance. Therefore, the correctness of these configuration parameters is assumed in our approach. The attestation of confidential configuration data to the global configuration database in the back-end (GRoT) is still required to verify the successful application of configuration data. That is, any malicious user that does not apply a configuration update must be detected by the GRoT. Since the second layer needs to attest configuration parameters, *property-based* attestation is used to attest the correctness of confidential configuration data.

VI. Evaluation

To show the technical feasibility of our proposed configuration component, the hardware demonstrator shown in Fig. 7 was realized. This demonstrator comprises two different controllers. An Infineon XMC4500 microcontroller that is

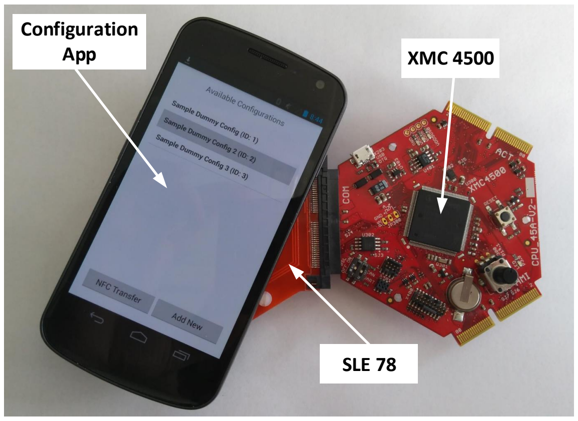


Fig. 7. Configuration component hardware demonstrator and Android device running the NFC configuration application.

used as the sensor controller in our approach and an Infineon SLE 78 [35] that is used as SE. As a mobile configuration device, we used an NFC-enabled Android smartphone. Using this demonstrator, configuration update times (reboot of the system not included) of about 200 ms can be achieved for configurations consisting of 5-10 configuration parameters including the necessary overhead imposed by the implemented security mechanisms.

A. STRIDE Threat Analysis

To highlight the achieved security level, a threat analysis was conducted that demonstrates the lists the threats (*T*) that can be mitigated by countermeasures (*C*) implemented our approach. Further, the threats are categorized by the STRIDE model (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege). Although we do not claim this threat analysis to be exhaustive, we think the listed threats represent the most relevant threats for a smart sensor configuration scenario. Regarding the threat analysis, we make the following assumptions: (i) We assume the global configuration database back-end is sufficiently secured against any kind of attack; therefore, it is considered as trusted entity. (ii) Also the SE used in our approach is considered as trusted entity. (iii) Other than that, we assume all other involved entities as being untrustworthy. **T1** (S, R, I, D, E): Adversary that is trying to perform remote attacks on the configuration interface. **C1**: Mitigated by using NFC which requires an adversary to be in close proximity to the smart sensor. **T2** (R, I): Adversary in close proximity is trying to eavesdrop or manipulate configuration packages. **C2**: Mitigated by using AE. **T3** (S, D): Adversary in close proximity is trying to use a captured configuration package to perform replay attacks. **C3**: Appropriate countermeasures are included in configuration package to mitigate these type of attack. **T4** (S, R, I, D, E): An adversary is able to inject malicious code or manipulated configuration data into the sensor controller. **C4**: When attested, the sensor controller is not able to unseal its network stack stored in the SE. Thus, the malicious smart sensor is blocked from accessing the network. **T5** (S, R, D): Malicious code in the sensor controller performs attacks (e.g. DoS) targeting other network devices or tries to replicate the malicious code to other devices. **C5**: When attested, the sensor controller is not able to unseal its network stack stored in the SE. Thus, the malicious smart sensor is blocked from accessing the network. **T6** (D): A malicious user tries to manipulate the functionality of a smart sensor by not applying a necessary configuration update. **C6**: The global configuration database back-end attests the configuration state of a smart sensor; therefore, not updated devices can easily be detected. **T7** (D): Adversary in close proximity tries to perform a DoS attack by continuously sending malicious configuration packages to the smart sensor. **C7**: The updates are rejected by the SE. Normal operation of the smart sensor is not impacted since the SE is powered through the mobile device's NFC field; therefore, no power required by the smart sensor is consumed. Also, all cryptographic operations to decide if a package needs to be rejected are performed at the SE, which does not impact the normal operation of the sensor controller. **T8** (S, T, R, I, D, E): Adversary with physical access to the smart sensor tries to perform physical and side-channel attacks to reveal confidential data such as key material or cryptographic algorithms. **C8**: The used SE mitigates physical attacks by implementing appropriate countermeasures. The security level is certified by CC.

VII. Conclusion and Future Work

In this paper we present a hardware-secured configuration approach based on NFC that is suitable for both confidential and non-confidential data alike. Our approach comprises (i) a component that can be included into future smart sensors as well as into legacy devices and (ii) a NDEF-based configuration protocol. The protocol includes information to prevent updates from malicious users and mitigates replay attacks. By allowing only NFC for configuration changes, the configuration interface is not exposed to remote attacks from the network. In addition, we also propose a two-layer configuration attestation architecture to attest the correctness of applied configuration updates. This architecture is capable of attesting non-confidential configuration locally using an SE as well as confidential configuration data remotely using a trusted global configuration database. The technical feasibility of our architecture is shown by means of a hardware demonstrator. In addition to that, the security properties are evaluated in a STRIDE threat analysis that highlights the increased security.

As future work we plan to investigate different methods to grant or deny network access for smart sensors regarding their trade-off between provided security level, and overhead.

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] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton, "Smart Objects as Building Blocks for the Internet of Things," *IEEE Internet Computing*, vol. 14, no. 1, pp. 44–51, 2010.

[2] G. Meijer, K. Makinwa, and M. Pertijs, *Smart Sensor Systems: Emerging Technologies and Applications*. John Wiley & Sons, 2014.

[3] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT Systems: Design Challenges and Opportunities," in *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2014, pp. 417–423.

[4] H. C. Pöhls, V. Angelakis, S. Suppan, K. Fischer, G. Oikonomou, E. Z. Tragos, R. D. Rodriguez, and T. Mouroutis, "RERUM: Building a Reliable IoT upon Privacy- and Security- enabled Smart Objects," in *Wireless Communications and Networking Conference Workshops (WCNCW), 2014 IEEE*. IEEE, 2014, pp. 122–127.

[5] S. Weyer, M. Schmitt, M. Ohmer, and D. Gorecky, "Towards Industry 4.0 - Standardization as the crucial challenge for highly modular, multi-vendor production systems," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 579–584, 2015.

[6] L. Schor, P. Sommer, and R. Wattenhofer, "Towards a Zero-Configuration Wireless Sensor Network Architecture for Smart Buildings," in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. ACM, 2009, pp. 31–36.

[7] J. Lee, B. Bagheri, and H.-A. Kao, "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18–23, 2015.

[8] S. Nastic, S. Sehic, D.-H. Le, H.-L. Truong, and S. Dustdar, "Provisioning Software-Defined IoT Cloud Systems," in *Future Internet of Things and Cloud (FiCloud), 2014 International Conference on*. IEEE, 2014, pp. 288–295.

[9] C. Perera, P. P. Jayaraman, A. Zaslavsky, D. Georgakopoulos, and P. Christen, "Sensor Discovery and Configuration Framework for The Internet of Things Paradigm," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 94–99.

[10] M. Steger, C. Boano, M. Karner, J. Hillebrand, W. Rom, and K. Römer, "SecUp: Secure and Efficient Wireless Software Updates for Vehicles," in *Digital System Design (DSD), 2016 Euromicro Conference on*. IEEE, 2016, pp. 628–636.

[11] 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.

[12] 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.

[13] 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.

[14] T. Ulz, T. Pieber, C. Steger, S. Haas, H. Bock, and R. Matischek, "Bring Your Own Key for the Industrial Internet of Things," in *2017 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2017, pp. 1430–1435.

[15] T. Rauter, A. Höller, J. Iber, and C. Kreiner, "Thingtegrity: A Scalable Trusted Computing Architecture for the Internet of Things," in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*. Junction Publishing, 2016, pp. 23–34.

[16] H. Tan, W. Hu, and S. Jha, "A TPM-enabled Remote Attestation Protocol (TRAP) in Wireless Sensor Networks," in *Proceedings of the 6th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*. ACM, 2011, pp. 9–16.

[17] P. Koeberl, S. Schulz, A.-R. Sadeghi, and V. Varadharajan, "TrustLite: A Security Architecture for Tiny Embedded Devices," in *Proceedings of the Ninth European Conference on Computer Systems*. ACM, 2014, p. 10.

[18] S. Saroiu and A. Wolman, "I am a Sensor, and I Approve This Message," in *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*. ACM, 2010, pp. 37–42.

[19] W. Arthur and D. Challener, *A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security*. Apress, 2015.

[20] C. Castelluccia, A. Francillon, D. Perito, and C. Soriente, "On the Difficulty of Software-Based Attestation of Embedded Devices," in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 400–409.

[21] C. Kil, E. C. Sezer, A. M. Azab, P. Ning, and X. Zhang, "Remote Attestation to Dynamic System Properties: Towards Providing Complete System Integrity Evidence," in *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*. IEEE, 2009, pp. 115–124.

[22] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla, "SCUBA: Secure Code Update By Attestation in Sensor Networks," in *Proceedings of the 5th ACM workshop on Wireless security*. ACM, 2006, pp. 85–94.

[23] A.-R. Sadeghi and C. Stüble, "Property-based Attestation for Computing Platforms: Caring about properties, not mechanisms," in *Proceedings of the 2004 workshop on New security paradigms*. ACM, 2004, pp. 67–77.

[24] B. Benyo, A. Vilmos, K. Kovacs, and L. Kutor, "NFC Applications and Business Model of the Ecosystem," in *Mobile and Wireless Communications Summit, 2007. 16th IST*. IEEE, 2007, pp. 1–5.

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

[26] 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.

[27] M. Rogawski and K. Gaj, "A High-Speed Unified Hardware Architecture for AES and the SHA-3 Candidate Grøstl," in *Digital System Design (DSD), 2012 15th Euromicro Conference on*. IEEE, 2012, pp. 568–575.

[28] P. Kocher, R. Lee, G. McGraw, A. Raghunathan, and S. Moderator-Ravi, "Security as a New Dimension in Embedded System Design," in *Proceedings of the 41st annual Design Automation Conference*. ACM, 2004, pp. 753–760.

[29] 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.

[30] 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.

[31] A. Vasudevan, E. Owusu, Z. Zhou, J. Newsome, and J. M. McCune, "Trustworthy Execution on Mobile Devices: What Security Properties Can My Mobile Platform Give Me?" in *International Conference on Trust and Trustworthy Computing*. Springer, 2012, pp. 159–178.

[32] M. Bond and R. Anderson, "API-Level Attacks on Embedded Systems," *Computer*, vol. 34, no. 10, pp. 67–75, 2001.

[33] R. Sailer, X. Zhang, T. Jaeger, and L. Van Doorn, "Design and Implementation of a TCG-based Integrity Measurement Architecture." in *USENIX Security Symposium*, vol. 13, 2004, pp. 223–238.

[34] 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.

[35] Infineon Technologies AG, "Infineon Chip Card & Security ICs Portfolio," 2015, accessed on 04/04/2017. [Online]. Available: http://www.infineon.com/cms/en/product/security-and-smart-card-solutions/security-controllers/sle78/channel.html?channel=5546d462503812bb015066c2d8e91745

[36] M. Hutter and R. Toegl, "A Trusted Platform Module for Near Field Communication," in *Systems and Networks Communications (ICSNC), 2010 Fifth International Conference on*. IEEE, 2010, pp. 136–141.

[37] Y. Yang, X. Wang, S. Zhu, and G. Cao, "Distributed Software-based Attestation for Node Compromise Detection in Sensor Networks," in *Reliable Distributed Systems, 2007. SRDS 2007. 26th IEEE International Symposium on*. IEEE, 2007, pp. 219–230.

[38] A. Seshadri, A. Perrig, L. Van Doorn, and P. Khosla, "SWATT: SoftWare-based ATTestation for Embedded Devices," in *Security and Privacy, 2004. Proceedings. 2004 IEEE Symposium on*. IEEE, 2004, pp. 272–282.

[39] Y. Mo, E. Garone, A. Casavola, and B. Sinopoli, "False Data Injection Attacks against State Estimation in Wireless Sensor Networks," in *Decision and Control (CDC), 2010 49th IEEE Conference on*. IEEE, 2010, pp. 5967–5972.

[40] C. Kwon, W. Liu, and I. Hwang, "Security Analysis for Cyber-Physical Systems against Stealthy Deception Attacks," in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 3344–3349.