A Secure Hardware Module and System Concept for Local and Remote Industrial Embedded System Identification

C. Lesjak, T. Ruprechter, J. Haid, H. Bock
Infineon Technologies Austria AG
Design Center Graz, Austria
{christian.lesjak, thomas.ruprechter, josef.haid holger.bock}@infineon.com

Eugen Brenner
Institute for Technical Informatics
Graz University of Technology, Austria
brenner@tugraz.at

Abstract

Smart maintenance constitutes an essential concept in Industry 4.0, where industrial devices report their maintenance status to remote back end systems and thus predictive maintenance can be intelligently scheduled and carried out locally at the affected device. This status data must be securely assignable to the claimed device identities when transmitted remotely. Furthermore, during the actual maintenance task, the service technician must be able to trustworthy identify the correct target device. Unfortunately, current systems typically lack cryptographic authentication and a secure storage for the required credentials, causing identity impersonation as a major threat.

In this paper we present a secure NFC-enabled hardware module for industrial embedded systems with a secure identity, enabling local identification by means of the proximity based contact-less technology Near Field Communication (NFC), and remote identification via a contact-based interface, thus helping to prevent device impersonation attacks, device clones and human errors on device identification. A proof of concept utilizing an Infineon security controller capable of elliptic curve cryptography demonstrates the concepts feasibility.

1 Introduction

Initiatives, like the German high-tech strategy “Industry 4.0” [8] and GE’s “Industrial Internet” [5] anticipate the fourth industrial revolution. This convergence of machines and data is characterized by interconnected cyber-physical systems. It affects not only the production area, but also includes logistics, distribution, customer service, infrastructure, facility services and more. Industry 4.0 is based on self-optimization, self-configuration, self-diagnosis, and intelligent work support.

A key concept of this technological advancement is smart maintenance [2], where the aim is to keep automation systems operational and to reduce unplanned downtime by intelligent, predictive maintenance management.

1. Monitoring includes the local or remote data acquisition about a device’s current state, e.g., the number of operating hours. In case of remote monitoring, this data is transmitted via a network such as the Internet, to a central back end system of the device vendor. For local monitoring, a technician present at the device acquires the information.

2. Updating includes hardware fixes or replacements, and system or application level software or configuration updates. Such maintenance tasks require physical presence of a vendor technician at the affected device.

Figure 1. The two central use cases necessary for device identification.

This is achieved by supporting authorized entities to remotely keep track of the status of industrial devices, and to carry out appropriate maintenance tasks on-site (locally) at a device. Henceforth, we consider smart maintenance to encompass two sides (extended from [2]):
Both sides, monitoring and updating, rely on an unambiguous and trustworthy identification of each single device, both for local technicians as well as on remote data transmission to a maintainer’s central back end system via some network connection.

Therefore, each device requires a unique identity to allow a vendor to keep track of all its deployed devices, their specific set-up (firmware, configuration, hardware modules, sensors) and maintenance status (total hours of operation, etc.). This identity ID_
D for device D is a string of information and unique in at least the context of a single device vendor. Furthermore, for a device identity to be trustworthy, cryptographic authentication is necessary. In accordance to [11], we use the term identification synonymously to entity authentication. Both concepts describe the process where a verifier carries out an authentication protocol with a prover. As a protocol outcome, the verifier will either accept the prover’s identity as authentic, or will reject the claimed identity and terminate any further interaction.

Figure 1 depicts the two essential use cases necessary in our industrial embedded system identification concept. In order to identify a device, it first needs an identity to be provisioned. In the provisioning use case (1), a device vendor assigns a device its identity. Furthermore, in the second use case, a device is being identified, typically while deployed and in operation at a plant. This can be carried out by a service technician who verifies a device’s identity while being close to the appliance (2a), using a portable client such as a tablet computer. Alternatively, a device identifies itself over a network (or any other long distance) connection (2b), to the remote, central back end system of the device vendor.

In our paper we make the following contributions to provide secured device identification:

- We propose an architecture that enables secured provisioning, and secured storage of device identification information on a dedicated secure hardware module, denoted the ID module (IDM) (Section 3).
- We present a system concept for secured local, as well as secured remote identification of an industrial device, and show how to integrate the therefore necessary ID module into an industrial device and its embedded host controller. (Section 4).
- We exemplify the feasibility of our ID module and system by means of a smart maintenance case study carried out on an actual proof of concept implementation (Section 5).

2 Background

2.1 State-of-the-art systems and challenges thereof

In general, we can classify state-of-the-art systems into two categories: (1) The identifier is stored in the device host controller, and is also written onto the device case. Here, the identifier is not cryptographically authenticated and can be manipulated or cloned arbitrarily. Furthermore, human errors on reading identifiers printed onto devices are possible. (2) The identifier and credentials for authenticating a device are stored in the device host controller. Using the credentials (e.g., private keys), the identifier can be authenticated, yet the credentials are not protected against being extracted from the host controller to allow impersonation.

Henceforth, we identify the following threats.

1. An unauthenticated device identity can easily be reused by imitators to produce cloned (counterfeit) devices. Furthermore, the device can be easily impersonated when transmitting data to the remote back end system of the vendor, as no cryptographic authentication procedure takes place.

2. In cases where the identifier is authenticated using cryptographic credentials stored inside the host controller, a malicious entity or competitor might still extract these secrets, in order to produce counterfeit goods. Also, an attacker might use extracted credentials to impersonate device identities when communicating with a vendor back end system.

3. The human factor of misreading device IDs might have wrong devices being reported to the vendor as broken or to be maintained, or a maintenance technician sent to service a specific device might service a wrong device.

For our system concept, we differentiate three distinct stages during a device’s life cycle, as depicted in Figure 2. During manufacturing, the firmware and vendor public key are initialized on the secure hardware module in its protected ROM, leading to an unprovisioned device. During the provisioning phase, an identity gets assigned to a device by uploading a digital certificate including the device’s identification information onto the ID module. In the operational phase, the module provides its identification services via its contact-based and contact-less interfaces. A device’s ID may be re-provisioned at any time, but only by the authorized vendor.

2.2 Related technologies

The proposed architecture of the ID module comprises three essential technologies:
A secure micro-controller for credential storage and cryptographic operations: A security micro-controller is a tamper-resistant integrated circuit (IC), such as used in latest smart cards for payment. A number of security features make the extraction of key material, such as cryptographic keys for ID authentication, extremely difficult. This secure hardware builds the hardware platform for the proposed ID module, in order to securely store necessary credentials and carry out the necessary authentication procedures. The security of such micro-controllers conforms to internationally standardized security evaluations, like Common Criteria EAL5+ (high). [1]

Near field communication: Near field communication (NFC, [14]) provides a short-range, wireless data and power transfer link, which is established automatically once two NFC devices are brought into close proximity (max. few centimetres). The technology is based on existing contact-less standards such as ISO/IEC 14443 A and B. The secure micro-controller, which encompasses the ID module, provides not only a contact-based interface (to the host controller or embedded system), but also such an NFC interface for contact-less, proximity based communication.

Cryptographic scheme: In the asymmetric-key elliptic curve cryptography (EC, [6]), a secret private key \( d \) of a given domain \( G \) corresponds to a public key \( Q \). A digital signature over a given data string, calculated using the private key, can be verified by any entity that knows this public key. This scheme is known as elliptic curve digital signature algorithm (ECDSA). A digital certificate binds a public key to the identity of the owner of the corresponding private key, using a digital signature by the issuer of the certificate. Such a digital certificate, signed and issued by the device vendor can protect the device identities from manipulation by unauthorized entities by making manipulations detectable upon signature verification. Thus, only the authorized vendor may provision device identities, and these device identities cannot be manipulated by unauthorized entities.

3 Architecture of the ID module

3.1 Device and ID module

A schematic of an industrial device is depicted in Figure 3. Inside the device case, it encompasses a host micro-controller unit (MCU) that controls a device’s operational function. This host MCU is connected via an interface controller to a network (or some other wire connected) to transmit maintenance data to a remote, central data acquisition server (vendor back end). The host controller is further connected via a contact-based interface to the ID module to utilize its identification services for remote identification. The ID module itself also exposes its identification services via a contact-less interface according to ISO 14443 to contact-less readers, thus enabling local identification via NFC.

![Figure 3. Illustration of the device, which houses the host controller and the ID module.](image)

![Figure 4. Illustration of the proposed ID module architecture and the thus necessary memory contents of the secure hardware module.](image)

3.2 Detailed architecture

In Figure 4, the detailed architecture of the ID module is depicted. The ROM of the module has been initialized during manufacturing phase, equipping it with the firmware and services for provisioning and identification, and the public key \( Q_\text{V} \) of the vendor.

The secure non-volatile memory (NVM) is initialized when a device ID gets provisioned into the ID module. A secret private key \( d_D \), which never leaves the secure hardware module, corresponds to the device public key \( Q_D \), which is part of the device certificate \( \text{Cert}_D \). The certificate contains also the device identity string \( ID_D \), and is protected by a vendor signature, calculated using the vendor’s secret private key \( d_V \) that corresponds to the vendor public key \( Q_V \). The fixed vendor public key \( Q_V \) is used for the device certificate verification upon provisioning.

3.3 Provisioning of the device identity onto a device

After the manufacturing phase, a vendor provisions a device, or more precisely a device’s ID module, with its identity, using the NFC interface and a suitable NFC reader device, denoted \( R \). From the manufacturing phase, the ID module has the root public key \( Q_V \) of the vendor.
pre-installed with its firmware. To provision an identity to a device using an NFC enabled client, the vendor needs to issue a certificate $Cert_D$ to the device that binds the desired device identity $ID_D$ to its public key $Q_D$. The provisioning procedure is defined as follows:

\[
R \rightarrow IDM : \quad N_R \quad \text{(1a)}
\]

\[
IDM : \quad \text{generate } \{Q_D, d_D\} \quad \text{(1b)}
\]

\[
IDM \rightarrow R : \quad Q_D, \text{Sig}_D(N_R, Q_D) \quad \text{(1c)}
\]

\[
R : \quad \text{verify Sig}_D(N_R, Q_D) \quad \text{(1d)}
\]

\[
R : \quad \text{generate Cert}_D = \{ID_D, Q_D, \text{Sig}_V, \ldots\} \quad \text{(1e)}
\]

\[
R \rightarrow IDM : \quad \text{Cert}_D \quad \text{(1f)}
\]

\[
IDM : \quad \text{verify Cert}_D \quad \text{(1g)}
\]

In provisioning step (1a), a random nonce $N_R$ supplied by the reader $R$, initiates the process. The ID module $IDM$ generates a new, random ECC key key pair (1b), composed of the private key $d_D$, and the public key $Q_D$. That way, both entities contribute random input to the protocol in order to prevent replay attacks. The public key $Q_D$, as well as a signature (using the just generated secret key) over public key and nonce, are replied (1c). The reader verifies the signature to check if the ID module is in possession of the corresponding private key $d_D$ (1d), and issues a certificate containing the device ID $ID_D$ and its public key that has just been generated (1e). The certificate is sent to the device’s ID module (1f). The ID module will only accept the certificate if it can successfully verify the signature, which only succeeds if the private key $d_V$ that corresponds to its pre-installed public root key $Q_V$, was used to issue the certificate $Cert_D$. Otherwise the process will either recover to the previous provisioned certificate, or the ID module will stay un-provisioned. Therefore, only the vendor itself is able to provision device identities, but no other entity that has no knowledge of the private vendor key $d_V$, required to issue the device certificate $Cert_D$.

### 4 System concept for local and remote identification using the ID module

Figure 3 depicts the whole system concept, including a local and a remote verifier. The local identification of the device is carried out via an NFC link between reader $R$ and the ID module $IDM$. The reader’s software knows the public vendor root key $Q_V$, which is necessary to carry out the proposed identification protocol. Remote identification is accomplished via the interface controller of the device’s host embedded system. The host microcontroller forwards the request to the ID module, which responds its identity and a cryptographic proof thereof.

#### 4.1 Local identification

For local verification of the identity, the local verifier is present at the device with a portable, NFC-enabled client device. The client directly communicates with the ID module via the NFC link. In order to assess the authenticity of the device ID, it must know the public vendor root key for certificate verification. The portable verifier device is denoted $LV$ (local verifier) in the following protocol:

\[
IDM \rightarrow LV : \quad \text{Cert}_D \quad \text{(2a)}
\]

\[
LV \rightarrow IDM : \quad \text{verify Cert}_D \quad \text{(2b)}
\]

\[
LV \rightarrow IDM : \quad N_{LV} \quad \text{(2c)}
\]

\[
IDM \rightarrow LV : \quad \text{Sig}_D(N_{LV}) \quad \text{(2d)}
\]

\[
LV \rightarrow IDM : \quad \text{verify Sig}_D \quad \text{(2e)}
\]

In the local identification process, the verifier receives the device’s certificate $Cert_D$ (2a), which contains the device ID, and verifies (2b) this certificate using the public vendor root key $Q_V$ and the issuer signature. This public key is preconfigured in the verification client and is publicly known. A random nonce $N_R$ then challenges the ID module, which it must sign (2c, 2d) using the private and secret key stored inside the ID module. A nonce [13], which means ‘used only once’, is a common technique to provide freshness in a protocol sequence to prevent replay attacks. The local client verifies (2e) the received signature using the device’s public key $Q_D$ extracted from the certificate previously received and verified in steps (2a–2b). If all of the above steps terminate successfully, the device ID can be considered authentic and integer, and the device has henceforth been successfully identified in a secured way.

#### 4.2 Remote identification

For remote verification of the device identity, the remote verifier is located at a remote and typically distant location, connected via a network (or some other long distance connection) to the device’s host embedded system. In order for the host embedded system (denoted $H$) to provide the authentication information to the remote verifier ($RV$), the host needs to forward the requests and responses between verifier and ID module. The protocol is defined as follows:

\[
IDM \rightarrow H : \quad \text{Cert}_D \quad \text{(3a)}
\]

\[
H \rightarrow RV : \quad \text{Cert}_D \quad \text{(3b)}
\]

\[
RV : \quad \text{verify Cert}_D \quad \text{(3c)}
\]

\[
RV \rightarrow H : \quad N_{RV} \quad \text{(3d)}
\]

\[
H \rightarrow IDM : \quad N_{RV} \quad \text{(3e)}
\]

\[
IDM \rightarrow H : \quad \text{Sig}_D(N_{RV}) \quad \text{(3f)}
\]

\[
H \rightarrow RV : \quad \text{Sig}_D(N_{RV}) \quad \text{(3g)}
\]

\[
RV : \quad \text{verify Sig}_D \quad \text{(3h)}
\]
In the remote identification process, the verifier receives the device’s certificate $\text{Cert}_D$ via the host controller (3a, 3b) and verifies it using the public vendor root key $Q_V$ (3c). The subsequent steps are in accordance to the local verification described in Section 4.1, except the fact that the communication is being relayed via the network and the host controller (3d-3e, and 3f-3g). If all of the verification steps (3c, 3h) terminate successfully, the device ID can be considered authentic and integer.

5 Case study and assessment

5.1 Proof of concept

Our proof-of-concept system is depicted in Figure 5. The respective industrial device is composed of a host micro-controller, mounted on the bottom of the demonstrator, and the secure ID module and its analog RF front end on the upper side. An NFC enabled Android handset is used as a client to locally identify the device.

Device and secure hardware module.

The secure hardware module is realized in firmware on an Infineon security controller IC. As a reference industrial embedded system, we use a Texas Instruments MSP430 micro-controller and the corresponding TI Launchpad development platform. An I2C bus connects the MSP430 (master) to the security micro-controller (slave). The ID module is powered by the host embedded system.

Identification infrastructure.

A client for local identity verification has been implemented on an NFC-enabled Android handset. This smart phone connects via NFC to the analog NFC front end of the security micro-controller. The NFC communication utilizes the NFC Forum reader/writer mode, where the client acts as the master initiating requests to the passive slave, and may optionally also power the slave.

Remote verification is carried out via a USB connection from the MSP430 to a Windows PC. A Windows application carries out the identification protocol with the ID module via the MSP430 micro-controller, which acts as an I2C master to the ID module.

The industrial device demonstrator and the local verification device are pictured in Figure 6. The left picture shows the actual demonstrator hardware, while the right picture depicts the user interface of the portable, local identification client after successfully identifying a device.

5.2 Security assessment

From our case study the following results have been gained.

Security of the ID module hardware and firmware. Both, (1) the secret and integer storage of the cryptographic credentials inside the ID module, and (2) the cryptographic processing, must be sufficiently protected against attacks, in order to fulfil the proposed entity authentication in a secure manner. For the prototype, an EAL5+ (high) certified security micro-controller was used as the hardware for the ID module, which sufficiently provides the aforementioned protection requirements.

Provisioning scheme. The provisioning scheme depicted in Section 3.3 can be carried out successfully by entities which know the private key that corresponds to the public key in the ID module’s vendor root certificate. Given a secure key handling by the vendor, the proposed scheme is protected against attacks, aiming for the purpose of device identity impersonation or device cloning. The nonce provided during the provisioning process, as well as the randomly generated ECC key pair by the ID module, make both involved entities contribute randomness to the protocol in order to prevent replay attacks.

Identification schemes. The local, contact-less identification and entity authentication implies proximity of the verifier device due to the implied proximity of NFC (a few centimeters at maximum). The nonce provided by the verifier is used to check that the ID module must always be capable of calculating a new and correct signature given the nonce and the corresponding device certificate. In order for an attacker to clone a device, it must get to know the private, secret key inside the device ID module in order to successfully complete the authentication procedure against a verifier.

The remote, contact-based identification and entity authentication is per definition carried out by a remote entity. An attacker that impersonates a device must again
get knowledge of the secret key of the targeted device it desires to impersonate, in order to successfully complete the authentication procedure.

Mitigated attacks and problems. First, device clones or counterfeits can be detected as such later. Each device has its unique ID and unique private key, which are stored inside the device ID module. In order to clone a device, the counterfeit device must be capable of successfully authenticating its ID upon local or remote identification. Yet this procedure can be carried out successfully if the private key is known. Henceforth, a verifier can be enabled to detect cloned devices as they cannot complete the identification protocol successfully. Second, the remote back end system can trust a device’s identity when communicating with the back end. Devices with an ID module are able to successfully authenticate against the remote verifier. Third, service technicians are protected from servicing wrong devices due to misidentification or reading errors, as their portable client for local identification automatically may compare the ID of the device in front of the technician with the ID of the device to be maintained.

Local identification on device failure. Even in cases where the device, or its host embedded system becomes broken for some reason, the ID module can be still in operation and accessed using the local verification client. This is achieved by using the NFC technology for communication, which does not only provide a data link into the ID module, but also powers the module. Thus, even without a functioning host embedded system, or on power failure, the ID module is still fully operational in local identification mode.

Single-step provisioning for multiple purposes. In traditional systems, a device ID must be both, stored inside the host embedded system for remote identification, as well as visually attached to the device case for visual, local identification. Using the proposed ID module, our solution serves the provisioned ID to both, local and remote verifiers. Furthermore, errors during provisioning, where the digital and the visual ID do not match due to misconfiguration, are efficiently encountered.

5.3 Industry relevance

In the context of Industry 4.0 and specifically smart maintenance, two aspects become ubiquitous. First, automated data processing requires trust of the involved entities and proper security. Our concept provides cryptographically authenticated device identities, which secure network identification and protect against counterfeits and clones. Second, mobile clients nowadays mainly present in consumer field, will likely become prevalent also in the industrial environment to increase process efficiency for maintenance services. Our secure ID module reduces human mistakes like misidentifying target devices. The automated processing of the ID allows to provide further contextual information on the mobile client, and works even in certain cases of device failures.

5.4 Alternative wireless technologies

NFC is characterized by its rather short operating distance of less than 10 cm under normal conditions. Competing wireless technologies provide vastly greater nominal ranges between 10 m and 100 m, such as Bluetooth, ZigBee and Wi-Fi. In a typical industrial setting, multiple devices will typically be in the range of a few meters. Therefore, a pairing or selection mechanism of the desired device to communicate with is necessary. With NFC, communication is established with only the desired device, to which the mobile client is in very close proximity, without any further manual pairing effort. From security perspective, NFC has a number of advantages. Regarding the confidentiality of the transfer of a device’s identity, the limited communication range of NFC requires adversaries to be rather close. Eavesdropping of NFC communication has been reported up to 2.4 m [17]. Albeit this is 24 times the nominal reading distance, an adversary would still require close proximity where he will likely be detected by the regular employee. Furthermore industrial sites typically already have some sort of physical access control in place, making eavesdropping rather infeasible. Second, regarding availability, an attacker again must get close to the targeted industrial device to disturb communication by e.g., jamming. Such an attack would be easily spotted by an actual employee identifying a device via NFC.

6 Related work

The work presented in [4] theoretically discusses secure identifiers and their initial bootstrapping process in the context of the Internet of Things (IoT). The authors classify secure device identifiers into four categories, where our presented approach conforms to their “assigned secure IDs”. A device’s ID is comprised of an individual private key, a corresponding public key, and a certificate to certify the public key and additional device related information by a trusted third party. The so called secure device ID module stores the private key protected from outside access, and a management and service interface exposes identification and authentication services. We advance their theoretical, network authentication approach by a second, contact-less interface for local identification; and a case study on an actual proof-of-concept implementation.

The idea of integrating a Trusted Platform Module (TPM) with embedded systems to attest their status via network has been presented various times, e.g., in [9], [10]. Yet, a TPM requires adherence to the full specification of the services and protocols as specified by the Trusted Computing Group (TCG), and aims at providing platform attestation. Our solution provides a lightweight design that allows device identification, without unnecessary overhead. No local device identification in our means is possible, as only a contact-based interface to the TPM is given in these related works.

In [7] and [16] an NFC interface provides local at-
tation of public terminals using a TPM, where the aim is to check the integrity of the target device using a portable client. Yet, the authors do not consider identification via a different, contact-based interface.

[15] introduces the theoretical concept of a trust anchor that protects the integrity and authenticity of a device’s communication in industrial networks. The authors propose different ways how to integrate this trust anchor to protect a device’s integrity, and to securely authenticate its transmitted data on the network. Again, the authors focus on the device integrity, and network-based identification, only.

Druml et al. [3] as well as Menghin et al. [12] equip an embedded system with an NFC interface, in order to enable communication between the embedded system and a portable, NFC-enabled client via this NFC link.

7 Conclusion and future work

Future industrial use cases such as smart maintenance require new means of security, which help to prevent device impersonation, device clones and human mistakes. Especially in interconnected, automated systems, where devices transmit status data to remote locations without human interaction, secure identification is inevitable.

This paper presents two procedures for industrial device identification, namely local and remote identification. We propose the architecture of a dual-interface ID module, which provides a secure storage for the necessary credentials. We also present a system concept how to integrate this module into an industrial device, and how to carry out the provisioning of device IDs by authorized entities only, such as the device vendor. For both, ID provisioning as well as identification and verification, the necessary procedures have been described and explained.

We exemplified the feasible implementation and integration of our ID module in a proof of concept system by means of a device’s host embedded system and an Infineon security controller. Successful identification through local and remote identification schemes have been carried out, using an Android client with an NFC interface, and a remote verifier represented by a Windows PC.

In future work, the device life cycle may be extended to suit more complex scenarios, such as where more parties are involved, e.g., a dedicated maintenance service provider. From a security perspective, a potential revocation of the vendor root certificate in case of compromise of the secret, private vendor root key, should be further considered, as industrial devices may be in use for many years or decades.

Acknowledgement

The authors would like to thank the Austrian Federal Ministry for Transport, Innovation and Technology as well as the ARTEMIS Joint Undertaking, which funded the ARROWHEAD project (file number ART-010000-2013-3) under the FP7-JTI programme.

References