

RESEARCH ARTICLE

A Generic Credential Management Model for Secure Field-Level Communication in IIoT Networks

JULIAN GÖPPERT¹, ANDREAS WALZ, AND AXEL SIKORA¹

Institute of Reliable Embedded Systems and Communication Electronics, Offenburg University of Applied Sciences, 77652 Offenburg, Germany

Corresponding author: Julian Göppert (julian.goeppert@hs-offenburg.de)

This work was supported in part by German Program for the Industrial Community Research (IGF) as part of the FieldPKI Project under Grant AIF/DFAM21752N. We acknowledge support by the Open Access Publication Fund of the Offenburg University of Applied Sciences.

ABSTRACT Increasing numbers of cyberattacks and stringent regulations necessitate cryptographic protection for field-level communication in Industrial Internet of Things (IIoT) networks. This requires device-specific credentials, comprising a private key, a public key certificate, a trust anchor, and optional authorization attributes for IIoT components. This paper proposes a generic and protocol-independent credential management model spanning the entire lifecycle of IIoT components, from manufacturing to disposal. The model's novel contributions include: 1) a credentialing entity orchestrating secure credential provisioning, renewal, and revocation, 2) dedicated control domain credentials enabling mutual authentication and authorization, and 3) lifecycle-oriented processes ensuring scalability and automation. These features bridge Information Technology (IT) and Operational Technology (OT) environments, satisfy security requirements (e.g., origin verification, protected initial credential provisioning, mutually authenticated and authorized subsequent credential management), and support resource-constrained systems. This enables secure and scalable credential management for diverse IIoT protocols and contextualizes existing approaches, enhancing trust and manageability in IIoT environments. The model is validated with a physical CANopen-FD-specific implementation, proving the feasibility for IIoT networks lacking native credential management. Formal security analysis using ProVerif proves robustness against impersonation, unauthorized access, and man-in-the-middle attacks under a Dolev-Yao adversary, addressing the lack of formal verification in existing standards.

INDEX TERMS Credential management, field-level, lifecycle, operational technology, public key infrastructure, security configuration.

I. INTRODUCTION

The Industrial Internet of Things (IIoT) enables interconnected *industrial automation* (IA) components, such as PLCs, sensors, and actuators, to execute real-time applications and exchange data via protocols like PROFINET [1], Open Platform Communication Unified Architecture (OPC UA) [2], or CANopen FD [3]. Increasing numbers of cyberattacks and regulatory requirements call for the authenticity-, integrity-, and confidentiality-protection of field-level communication, which otherwise would risk production

downtime, environmental harm, or equipment damage in safety-critical systems [4]. Symmetric cryptographic protection secures exchanged data, but requires shared security parameters (e.g., symmetric keys) established through *authenticated* and *authorized key agreements* using *public key certificates* [1]. This necessitates robust credential management—encompassing private keys, public key certificates, trust anchors, and authorization attributes—across the lifecycle of IA components, from manufacturing to disposal.

We propose a generic (i.e., protocol-independent) model of processes and communication protocols to answer our main research question: *How can credentials be managed*

The associate editor coordinating the review of this manuscript and approving it for publication was Chi-Yuan Chen¹.

for field-level devices in IIoT networks throughout their lifecycle? On the one hand, the motivation for this generic model is that it can be universally transferred to protocols or technologies that currently do not feature credential management mechanisms, enabling secure communication. On the other hand, the model contextualizes existing credential management concepts of diverse protocols (e.g., PROFINET, OPC UA, Common Industrial Protocol or IEC/IEEE 60802). Moreover, existing standards lack a unified, protocol-agnostic lifecycle-spanning approach, which our generic model addresses. The challenges of developing the generic model at hand are: developing the model such that it meets the requirements formulated in Section IV, finding the least common denominator of existing approaches and describing the commonalities in a technology-independent way. This paper consolidates and completes our prior partial contributions on formal analysis, implementation, and performance evaluation [5], [6], [7], [8], presenting the full generic model for the first time.

Its novel contributions include:

- 1) a *credentialing entity* orchestrating credential management processes,
- 2) *dedicated control domain credentials* forming the underlying basis to execute the management processes in a mutually authenticated and authorized manner, and
- 3) *credentialing processes* performed throughout the lifecycle of IA components within IIoT environments.

To rigorously analyze the model's security and address the lack of formal security analyses in existing standards, its security properties are proven using the symbolic protocol verifier ProVerif [9]. Validated via a CANopen-FD-specific implementation on a physical demonstrator, the model enhances security and offers a standardized framework for IIoT applications.

The paper begins with definitions and background (Section II), reviews related work (Section III), and outlines the scope, threats, and requirements (Section IV). It then details the credentialing entity (Section V), dedicated credentials (Section VI), and management processes (Section VII). The formal verification of the model's security properties (Section VIII) and a CANopen-FD-specific implementation follow (Section IX), before concluding the paper (Section X).

II. DEFINITIONS AND BACKGROUND

In this section, we define a credential object and describe its role in unilateral and mutual authentication and authorization, critical for secure field-level communication.

Credential objects, which are the managed artifacts on IA components, lack a universal definition in literature and standards. Here, we propose a comprehensive definition, noting similarities with standards like PROFINET [1] or IEEE 802.1AR [10]. A credential object (see Fig. 1) comprises up to four elements: a *secret* (S), a *certification path* (C), a *trust anchor* (T), and *authorization attributes* (A). Different instances of credential objects exist based on their elements. For example, a credential object may include

only a secret and certification path for authenticating to another entity, or only a trust anchor and authorization attributes to validate another entity (see Fig. 1). PROFINET defines a credential in its attribute model as a triplet of secret, certification path, and trust anchor with authorization attributes embedded in X.509 end entity (EE) certificates, aligning closely with our approach [1]. In the IEEE 802.1AR standard for secure device identities, a credential is defined as the pair of secret and certification path, while trust anchors are used to validate the credential [10]. In the following, we explain the elements of our credential object definition:

- A *secret* is the private key of an asymmetric key-pair (e.g., a 256-bit secp256r1 elliptic curve key), known only to the legitimate owner and used to prove the own identity.
- The *certification path* includes an EE certificate and, potentially, intermediate certification authority (CA) certificates. The EE certificate binds the public key (corresponding to the secret) to identity information (e.g., serial number, IP address, or device name) via the issuer's signature. Most IIoT protocol standards that feature certificate-based authentication, such as PROFINET, OPC UA, and IEC/IEEE 60802, use X.509 certificates per IETF RFC 5280 [11]. The IA component's identity information, for example, properties that are known by the device manufacturer at production time (e.g., a serial number) or locally significant identity information (e.g., IP addresses or names) can appear as subject names or subject alternative name extensions. Additionally, X.509 certificates contain a validity period, which makes them forward-looking assertions that may cease to apply in the future. Therefore, mechanisms to invalidate them are required. Furthermore, since it is good practice to limit their validity period, inherently their renewal is required.
- The *trust anchor*, often a certificate or a raw public key, is used for the validation of certification paths of other entities. IA components reject peer connections if the validation of the certification path with the configured trust anchors fail.
- *Authorization attributes* validate an entity's permissions, particularly for credential management (e.g., allowing an entity to manage credentials on an IIoT component). In PROFINET, attributes are encoded as extensions in EE certificates, assigning roles like *CredentialManager* to entities entrusted with the management of credentials, so-called Security Infrastructure Handlers (SIHs) [1].

Credential objects enable unilateral and mutual authentication and authorization. In unilateral authentication, entity A sends its certification path to entity B, undergoing three steps: (A) proof of possession (PoP) of the secret, (B) certification path validation using the trust anchor, and (C) authorization validation, all of which must be executed successfully (see Fig. 1). This order may vary by protocol. Mutual authentication and authorization involves

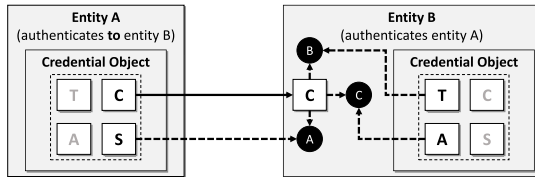


FIGURE 1. Unilateral authentication and authorization using credential objects comprise the steps described as (A) proof of possession, (B) certification path validation, and (C) authorization validation.

both entities independently performing unilateral authentication and authorization.

III. RELATED WORKS

Related works on credential management for field-level devices in IIoT networks stem from research papers and industrial communication standards. Our detailed survey [5] provides a comprehensive baseline by investigating 2042 publications, analyzing proposed entity architectures, supported functions, stakeholder involvement, and lifecycle considerations. This section summarizes key contributions of [5] and derives insights relevant to our work.

Some relevant standards address credential management for IIoT protocols, notably OPC UA [2], PROFINET [1], Common Industrial Protocol (CIP) Security [12], and IEC/IEEE 60802 [13].

PROFINET [1] defines the supply and management of security configuration, including site-specific certificates for PROFINET components. The approach comprises a Security Infrastructure Handler (SIH) that pushes certificates, private keys, and trust anchors to components via a PROFINET-specific protocol.

In OPC UA, a *GlobalDiscoveryServer* (GDS) manages certificates, certificate revocation lists (CRLs), and trust lists, supporting a push or pull behavior (i.e., a GDS pushes credentials to OPC UA devices or OPC UA devices pull credentials from a GDS) [2]. Moreover, there is an onboarding mechanism described for OPC UA devices [14].

CIP Security employs a *Certificate Management Object* (CMO) for X.509 certificate management, enabling devices to generate certificate signing requests for CA issuance [12].

IEC/IEEE 60802, a time-sensitive networking (TSN) profile for IIoT, uses NETCONF/YANG over the Transport Layer Security (TLS) protocol to manage certificates, keys, and trust anchors, with TSN Domain Management Entities (TDMes) acting as NETCONF clients [13], [15].

These standards describe their protocol-specific credential management and lack the description of a generic model as they vary in protocol representation, communication paradigms (push vs. pull), supported processes, and lifecycle scope. While remaining protocol-agnostic, our work develops a generic model, abstracting the following commonalities: entity architectures (Section V), credential types distinguishable by their purpose and utilization (Section VI), and comprehensive lifecycle coverage (Section VII).



FIGURE 2. Considered lifecycle of an IIoT component.

A number of research studies also explores credential management in IIoT. Tailored approaches exist for PROFINET [16], OPC UA [17], [18], [19], and MQTT-SN [20]. Some focus on initial commissioning (onboarding) [21], [22], while others adapt protocols or certificate formats for resource-constrained IIoT components, proposing lightweight certificate profiles [23], [24] or revocation mechanisms [24], [25], [26]. Despite these efforts, to the best of our knowledge no work proposes a generic model for continuous credential management across the entire lifecycle of IIoT components and allowing the contextualization of existing approaches, a gap our model addresses.

IV. SCOPE, THREAT MODEL, AND REQUIREMENTS

This section defines the scope for our generic credential management model, outlines the considered threat model, and derives security, functional, and usability requirements to mitigate threats and address IIoT system characteristics, contrasting IIoT with classical IT systems. These requirements are referenced in the remainder of the paper once they are met.

A. SCOPE

The main research question of this paper is: *How to manage credentials on IA components throughout their lifecycle?* We focus on creating the preconditions to continuously enable mutual authentication of IA components (as per Section II). However, the further utilization of the credentials (e.g., securing application data exchange) is out of scope.

The model is protocol-independent, designed for universal implementation across IIoT protocols, including those lacking credential management (e.g., CANopen FD), while reflecting approaches in standards like PROFINET and OPC UA (Section III). Detailed technology mappings are covered in our prior work [5].

The lifecycle of an IIoT component aligns with the reference architecture model for Industry 4.0 (RAMI4.0) [27]. Post-manufacturing, the component is shipped to its owner or operator (O/O) for commissioning comprising the installation, integration, and configuration. The operational phase follows, with potential maintenance phases for updates or reconfiguration. The component may transition to decommissioning, re-enter commissioning (e.g., resale), or proceed to disposal (see Fig. 2).

B. THREAT MODEL

This subsection expounds the threat model and describes the attacks that shall be mitigated by the realization of our model. We adopt the well-founded threat model from the OPC UA security specification [2], assuming attackers

can eavesdrop, block, modify, store, and replay messages in O/O's IIoT networks [22]. This threat model aligns with the Dolev-Yao adversary assumed in our formal security analysis (Section VIII). Additionally, attackers may introduce malicious components, mimicking credible manufacturers, to enable product piracy or infiltrate O/O's sites [28]. After a successful infiltration, such components, if equipped with local credentials, could join trusted networks, enabling further attacks. Additionally, without enforcing authentication and authorization in the IA components, an attacker, potentially acting through an infiltrated device, may impersonate entities that are capable of managing credentials on legitimate devices [18]. This could lead to harm on an even larger scale. For our following elaborations, we assume that credentials are persistently stored and used in secure element components that render physical and side-channel attacks infeasible. Moreover, for the sake of simplicity, we do not further elaborate on the trust relations between human users and credential management systems.

C. REQUIREMENTS

Based on the scope and threats, we derive requirements, drawing on IEC/IEEE 60802 security guidelines [13], to ensure secure and practical credential management in IIoT systems. The requirements are summarized in Table 1.

TABLE 1. Security (S), functional (F), and usability (U) requirements.

#	Type	Requirement description
1	S	Verify IA component origin
2	S	Unilaterally authenticated initial credential provisioning
3	S	Mutually authenticated and authorized subsequent management
4	F	Mediation between the IT- and IIoT-world
5	F	Delegation of management tasks
6	F	Complete IIoT component lifecycle coverage
7	U	Allow IA component replacement without manual management

1) SECURITY REQUIREMENTS

First, an O/O must verify a component's origin before including it in the *circle of trusted entities* (**req.1**), mitigating product piracy and infiltration by untrusted devices.

Second, initial credential provisioning must be unilaterally authenticity-protected, i.e., the O/O ensures that the actually provisioned component matches the expected (**req.2**).

Third, post-initial O/O credential provisioning, all future management processes (provisioning, renewal, removal, revocation) must be mutually authenticated and authorized, allowing entities to verify counterparts' authenticity and permissions (**req.3**). This means every entity provisioned or updated with further credentials shall be able to decide whether to accept the management or not, based on the authenticity and the authorization attributes of the communication counterpart. Likewise, every entity that provisions entities with credentials shall be able to decide whether to perform management procedures or not, i.e., before credential management, it must be verified whether the IA component belongs to the *circle of trusted entities* of an O/O.

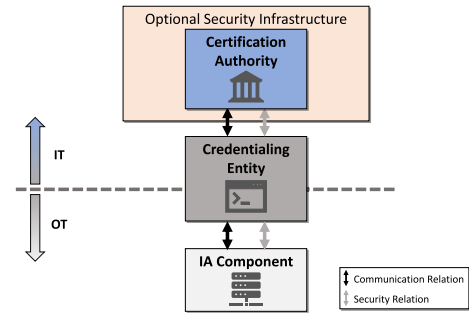


FIGURE 3. The reference architecture comprises four entities: an IA component, a CE, a CA, and an optional security infrastructure.

2) FUNCTIONAL REQUIREMENTS

Next to the security requirements, additional functional requirements are derived addressing the characteristics of IIoT networks.

First, in typical IT security architectures, certificates are issued by CAs. However, IA components likely only support the syntax and semantics of their IIoT protocol, whereas CAs likely only support interfaces from IT systems (e.g., TCP/IP-based protocols). Therefore, the solution shall provide means to translate and mediate between CAs from the IT-world and IA components from the IIoT-world (**req.4**).

Second, some IA components may only support a reactive communication paradigm (e.g., a PROFINET IO Device reacting upon the communication of a PROFINET IO Controller) and would thus never take on the active role of contacting a CA. Also, CA infrastructures likely only support reactive communication paradigms and only issue certificates upon request. Therefore, the model shall support the delegation of management tasks (**req.5**).

Third, the model must span the entire IIoT component lifecycle, not just considering the initial supply of credential object, but also covering the renewal, removal, and revocation of those (**req.6**).

3) USABILITY REQUIREMENTS

Lastly, we derive a non-functional requirement. The model shall allow the replacement of an IA component without manual security operations potentially only performed by security experts (**req.7**). Especially unplanned IA component replacements that cause production downtime are time-critical and should therefore be separated from manual and potentially time-consuming credential management decisions.

V. CREDENTIALING ENTITY

This section introduces the novel key element, namely a *credentialing entity* (CE), outlines its role in an architectural overview, and presents different communication paradigms.

Since IA components likely only support the syntax and semantics of their protocol (e.g., PROFINET [1]) and CAs likely only support interfaces from IT systems, a CE shall be an intermediary between CAs and IA components.

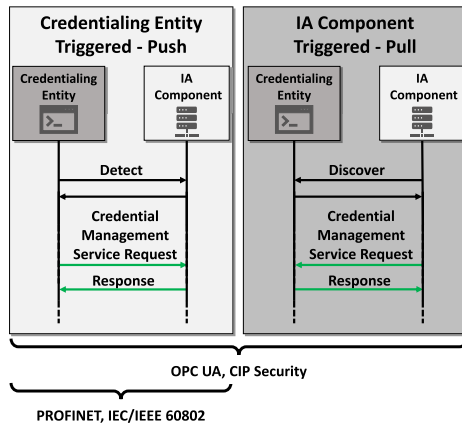


FIGURE 4. Communication paradigm variants of CEs and IA components and their support by standards.

Therefore, the CE helps to bridge the gap between the IT and the IIoT world, satisfying *req. 4*. Consequently, the reference architecture comprises four entities (see Fig. 3): the IA component to be provisioned with credential objects; the CE mediating between the CA and IA component; the CA, issuing certificates and revocation information; and an optional security infrastructure, hosting asset management or additional CAs. Communication and security relationships exist between the CE and IA component, and between the CE and CA. Multiple CEs and CAs may coexist at an O/O's IIoT site.

Depending on the protocol, a CE may support different communication paradigms for credential management. In the *push model*, the CE actively sends credential objects, revocation information, or key generation requests to IA components, which then react accordingly (e.g., PROFINET SIH [1] or IEC/IEEE TSN TDME [13]). In the *pull model*, the IA component actively requests credentials or revocation information from a reactive CE (e.g., OPC UA GDS [2]). Fig. 4 shows both paradigms including exemplary information flows and assigns their support by standards. Following these paradigms, the CE can be assigned credential management tasks, allowing it to play active push-like and reactive pull-like management scenarios, thus satisfying *req. 5*.

The CE, as a logical entity, can be integrated into various physical forms including non-IIoT-native forms. It may reside on a dedicated device (e.g., a tablet or laptop [16]) operated by a human for 1:1 communication with an IA component ensuring that it communicates with the expected component. Alternatively, the CE can be co-located on an IA component for autonomous operation. CAs can be hosted on external devices accessible via networks or integrated as a constituent of an IA components. The combination of autonomously acting vs. human-operated and internal vs. external CA allows four possible cases (see Fig. 5). Human-operated CEs are particularly appropriate for manual processes like the initial provisioning, removal, and revocation of credentials.

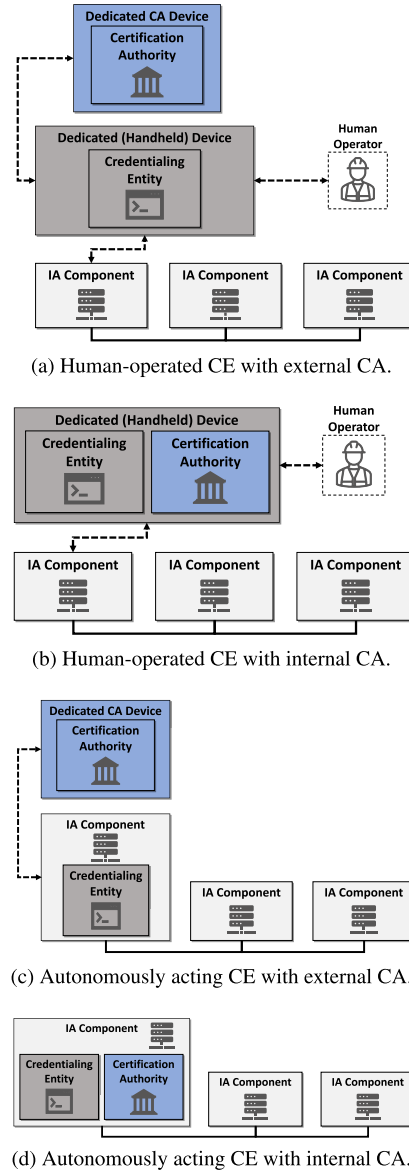


FIGURE 5. Integration options of CEs and CAs in IIoT environments.

Autonomously acting CEs may take on tasks like the subsequent provisioning or update of credentials (cf. *req. 7*). Internal CAs are less dependent on the accessibility of O/O operated or even company external CAs, but may impose a high-value target for attackers.

One O/O plant may comprise multiple CEs potentially in different integration options, e.g., one human-operated CE with an external CA for *taking* and *abandoning possession* and one autonomously acting CE with an integrated CA per automation system to impose the least dependencies on communication partners external to the automation system. As CEs are also considered as high-value targets, protection against unauthorized access is required, potentially through multi-factor authentication or restricted authorization to manage specific IA components via hierarchical PKI sub-CAs.

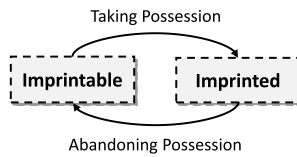


FIGURE 6. Taking possession requires the IA component to be in an imprintable state.

VI. DEDICATED CREDENTIALS

This section introduces the second novel key element, *dedicated credentials*, classified into three types: manufacturer, control domain, and application. The type discrimination of a credential object is not necessarily given by the properties of its elements, i.e., it is not always possible by the inspection of a credential object's content to decide what type of credential it is, as there may not be an explicit marking as application, control domain, or manufacturer credential. Rather, the classification as a particular type arises from its purpose. Below, we outline each type's purpose and role in meeting security and usability requirements.

Manufacturer credentials are issued and imprinted on IA components by the manufacturer before shipment to an O/O. Typically comprising a secret and certification path, they enable the O/O to unilaterally authenticate the component using the manufacturer's trust anchor (see Fig. 1, where entity A is the IA component and entity B is a CE). This verifies the component's origin and that the expected component instance matches the actual, mitigating rogue manufacturer threats and satisfying *req. 1*. The understanding of a manufacturer credential could also be widened to a supply chain credential reflecting multiple stages of the supply chain of an IA component, and thus multiple manufacturer credentials may be incorporated by one IA component.

Control domain credentials, the key innovation, are issued by the O/O when receiving a component and after optional manufacturer credential validation, during a process termed *taking possession*. They serve two purposes, detailed in the following:

- 1) enabling mutual authentication and authorization between CEs and IA components, answering whether a CE should provision credentials and whether a component should accept management (satisfies *req. 3*)
- 2) due to mutual authentication and authorization, the decision whether to perform subsequent management processes can automatically be taken, enabling to abstain from manual operations (satisfies *req. 7*).

The additional intermediate layer of control domain credentials enables mutual authentication and eliminates reliance on unilateral authentication using manufacturer credentials or no validation at all if manufacturer credentials are not present, enhancing security and usability in IIoT networks (cf. *req. 3*). Control domain credentials bind IA components to the O/O's domain. This aligns with the *resurrecting duckling* model [29], where components after transitioning

from an imprintable to an imprinted state only obey the O/O (see Fig. 6). Imprinting credentials from another domain requires a reversion to the imprintable state only possible upon the O/O's authorization.

Next to meeting security requirements, control domain credentials also have an impact on the usability. CEs can mutually authenticate IA components provisioned with control domain credentials and can therefore automatically decide whether to further provision them with application credentials. In time-critical situations like the replacement of a component, devices already taken into possession (e.g., temporarily stored as a spare part) can skip manual and therefore potentially time-intensive security procedures, thus satisfying *req. 7*.

Application credentials, issued by the O/O, support purposes like end-to-end secure application data exchange. From the perspective of our generic model, their exact purpose is out of scope. Instead, our goal is to describe a model that enables the management of application credentials aligning with the scope in Section IV.

VII. CREDENTIALING PROCESSES

This section combines the two novel elements, the credentialing entity from Section V and the dedicated credentials introduced in Section VI, to describe *credentialing processes* for managing control domain and application credentials in IIoT networks. We detail provisioning (including *taking possession*), renewal, removal, and invalidation, addressing all lifecycle events of IA components.

A. PROVISIONING

Provisioning processes equip IA components with control domain or application credentials. Since the presence of manufacturer credentials shall not be seen as a mandatory precondition of the model, but rather as a possible feature to safeguard the origin of an IA component, the provisioning of control domain credentials (*taking possession*) can occur either with unilateral authentication using a manufacturer credential or without shared security means. The final state of both credentialing processes shall be the same: an IA component that was in factory default state now holds a control domain credential from the domain of the O/O.

1) TAKING POSSESSION WITHOUT SHARED SECURITY MEANS

Taking possession without shared security means occurs after an O/O receives an IIoT component, transitioning it from a factory default to an imprinted state holding a control domain credential (per the *resurrecting duckling* model [29]). In a first step, a secret is generated, preferably by the component for optimal security, though resource-constrained components may rely on a CE or CA to generate it. A certificate signing request (CSR), incorporating the public key and identity information, is created and sent to a CA via a CE. The CA issues a certificate, which, along with the trust anchor (and potentially the secret, if not self-generated), is forwarded to the component by the CE.

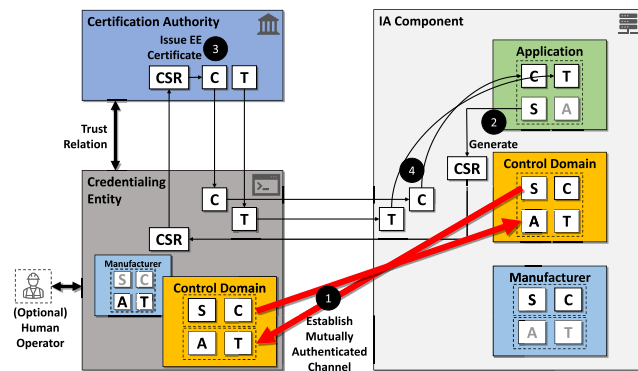


FIGURE 7. The provisioning of an application credential comprises the establishment of a mutually authenticated and authorized channel using the control domain credentials (1, wide arrows), generating a secret and a corresponding CSR (2), which is forwarded to a CA via a CE, the issuance of a certificate (3), and the imprinting of the certificate and corresponding trust anchor on the IA component via the CE (4).

Due to the lack of authentication means, man-in-the-middle attackers could impersonate an IA component as there is no way to establish an authenticated channel between an IA component and a CE. Therefore, taking possession without manufacturer credentials must proceed in a secure environment (e.g., through direct interaction, not over the network with multiple participants, but using physical proximity in a 1:1 connection between a human-operated CE and the IA component [16]).

2) TAKING POSSESSION WITH MANUFACTURER CREDENTIALS

Taking possession with manufacturer credentials follows a similar process but leverages unilateral authentication of the IIoT component by a CE, satisfying *req. 1* and *req. 2*. This enables remote execution over networks as manufacturer credentials verify the component's origin. For example, an engineer located in an engineering office can take possession of an IA component that is already installed in some system and reachable over a network connection. Standards like PROFINET, OPC UA, and IEC/IEEE 60802 use Initial Device Identifiers (IDevIDs) defined in IEEE 802.1AR [10] for this purpose. However, in both taking possession scenarios, IA components lack mechanisms to authenticate CEs, relying on Trust on First Use (TOFU) and allowing initial configuration by any initially connected CE. A 1:1 connection in physical proximity circumvents the risk of an attacker taking possession of a device.

3) PROVISIONING OF APPLICATION CREDENTIALS

Post *taking possession*, control domain credentials enable a mutually authenticated and authorized channel for provisioning application credentials (see Fig. 7). The process mirrors control domain credential provisioning: a secret and CSR are generated, forwarded to a CA via a CE, and the issued certificate and trust anchor are imprinted. In PROFINET, control domain credentials (termed LDevID-Generic) secure

the provisioning of application-specific credentials (termed LDevID-PN), with SIHs verified for *CredentialManager* role authorization [1].

B. RENEWAL

Typically, it is good practice that certificates have a limited lifetime, bounded by an expiration date [11]. Furthermore, changes in the certificate (e.g., changing authorization attributes that are encoded as role information in a certificate) throughout the lifecycle of an IA component [11] motivate the need to renew certificates. Application and control domain credential renewal follows the provisioning process, using control domain credentials for mutual authentication and authorization (see Fig. 7) with an option for automatic execution. Manufacturer credentials, designed as permanent identifiers (e.g., IEEE 802.1AR IDevIDs [10]) have per se no expiration date. Therefore, the proposed credentialing model does not comprise a dedicated process that enables the renewal of a manufacturer credentials.

C. REMOVAL

To avoid misuse, O/Os do not want to give a component with their credentials away. Therefore, decommissioning IIoT components for scrapping or resale motivates the introduction of the credentialing process to remove credential objects from IA components.

Both application and control domain credential removal use mutual authentication via control domain credentials. Removing a control domain credential, termed *abandoning possession*, reverts the component to an imprintable state. Since manufacturer credentials serve as permanent identifiers, their removal is conceptually forbidden.

D. INVALIDATION AND REVOCATION

By their nature, certificates are forward looking assertions i.e., from the issuance until the expiration date. However, certificates may become invalid before expiration e.g., due to private key compromise or device replacement. Furthermore, to avoid misuse, when an O/O abandons possession, an IA component's certificate shall not be valid any longer. A certificate shall only be regarded as valid until the expiration date is reached or until it is invalidated, whichever occurs first. This motivates the introduction of a credentialing process to invalidate certificates before their expiration date.

1) MODEL OVERVIEW

The underlying model, serving as a means to simplify the following considerations, consists of a security infrastructure (potentially hosting one or more CAs) that issues certificates and trust anchors, a subject provisioned with a certificate, and a certificate validator provisioned with a trust anchor. While performing certificate-based authentication, the subject's certification path is validated by the validator (see Fig. 8). Moreover, other inputs originating from the certificate subject or maintained on the validator itself may also be considered, depending on the implemented invalidation mechanism.

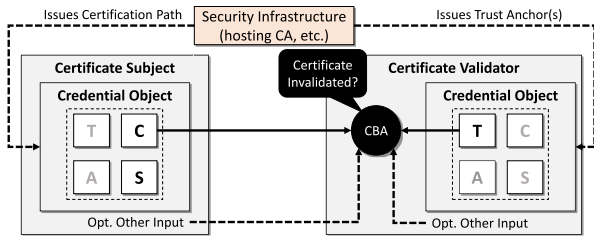


FIGURE 8. Certificate validators perform certificate based authentication (CBA) using the corresponding trust anchor and, depending on the invalidation mechanism, optional other inputs either maintained on the validator itself or inputs from the subjects.

In the most general sense, certificate invalidation shall be defined as a means that ensures that certificate validation fails. Whereas certificate revocation shall be defined as *active* and *selective* invalidation of certificates within a PKI hierarchy. Consequently, certificate revocation is only one approach of certificate invalidation. Conceptually, invalidation and revocation both comprise an invalidation *request*, triggering the security infrastructure to invalidate a certificate, and the subsequent certificate invalidation *enacting*, leading to the consequence that certificate validators regard certificates as invalid. The following elaborations focus on the certificate invalidation *enacting*.

2) CERTIFICATE REVOCATION

The first approach to realizing revocation is to maintain revocation information in certificate validators independent of certificate validation attempts. This requires *persistent storage* for e.g., Certificate Revocation Lists (CRL) [11] or certificate whitelists [30]. The second approach to realizing revocation is retrieving revocation information just-in-time with each certificate validation attempt. This requires an always online responder entity e.g., the Online Certificate Status Protocol (OCSP) [31]. The third approach to realizing revocation is to provide short-lived revocation information, i.e., non persistent information along with the certificate. For example, the certificate subject not only provides its certificate but a verifiable and recent claim by the CA that the certificate is still valid. Those claims are combined with time information, requiring secure and synchronized time information in certificate validators for the claim validation (e.g., OCSP stapling [32]).

3) CERTIFICATE INVALIDATION

The first approach to realizing invalidation but avoiding revocation is the use of short-lived certificates. This requires secure and synchronized time information in certificate validators. The second approach to realizing invalidation but avoiding revocation is to perform a roll-over of the current PKI. The basic idea behind this approach is to invalidate a certificate by removing the (intermediate) CA certificate it is signed with. This is applied (potentially multiple times) until the invalidated certificate is no longer

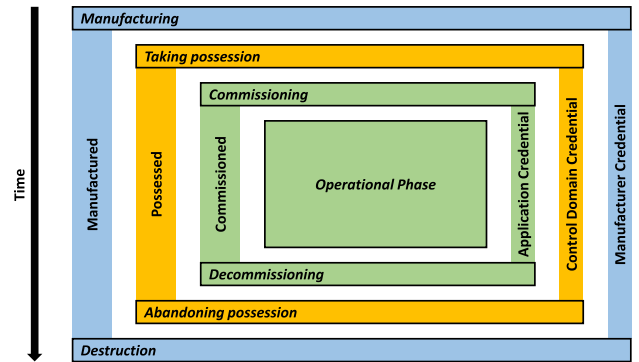


FIGURE 9. The combination of introduced credential types and credentialing processes results in this reference lifecycle of an IA component.

used as a trust anchor by all relevant certificate validators (i.e., no certificate validator uses a certificate as a trust anchor that is higher in the PKI hierarchy). However, this typically requires significant process overhead for each revocation event.

To conclude, all of the presented approaches come with conceptual elements that are undesirable in automation environments: persistent storage or secure and synchronized time information in certificate validators, a responder entity that is always reachable by certificate validators or, significant process overhead for each revocation event. A CE can serve as the direct interaction partner for to provision revocation information for certificate validators on the field-level. However, details depend on the particular protocol, technology, and, potentially also the requirements for a concrete system, which may also require blends of the above-mentioned strategies.

E. LIFECYCLE OVERVIEW

This subsection combines the credentialing processes to define a reference lifecycle for IA components in IIoT networks that governs credential management for IA components from manufacturing to disposal (see Fig. 9), satisfying *req. 6*.

During *manufacturing*, manufacturer credentials (e.g., IEEE 802.1AR IDevIDs [10]) are embedded as permanent identifiers. Upon arrival at an O/O site, *taking possession* provisions a control domain credential, transitioning the component to an imprinted state per the *resurrecting duckling* model [29]. In *commissioning* and during the *operational phase*, application credentials are provisioned, renewed, or removed via mutually authenticated channels. *Maintenance* involves credential updates, while *decommissioning* triggers *abandoning possession*, removing control domain credentials and invalidating others e.g., via Certificate Revocation Lists (CRLs [11]). Components may re-enter *taking possession* (e.g., resale) or proceed to *disposal*. The model supports events like company takeovers by reprovisioning credentials.

VIII. FORMAL SECURITY ANALYSIS

This section presents the formal security analysis of the credentialing processes' security properties using ProVerif [9], a state-of-the-art symbolic protocol verifier, to ensure the model satisfies *req. 1*, *req. 2*, and *req. 3*. We model certificate-based authentication and authorization for taking possession and automated application credential provisioning, proving robustness against impersonation, unauthorized access, and man-in-the-middle attacks under a Dolev-Yao adversary [9] as assumed in our threat model (Section IV). The Dolev-Yao adversary assumption abstracts cryptographic primitives as perfect, enabling scalable verification. Please note that this idealization overlooks probabilistic attacks, implementation flaws, or physical tampering. Therefore, concrete protocol realizations require further analysis of the selected algorithms, secure implementations, and physical properties which are not at focus in our paper (cf. Section IV). The ProVerif analysis confirms the model's security guarantees for IIoT networks and addresses the lack of formal verification in existing standards.

Detailed elaborations on the formal security analysis including the full ProVerif script are available in a separate publication [8], [33].

The ProVerif model comprises three entities: an IA component provisioned with a manufacturer credential, a credentialing entity managing control domain and application credentials, and a certification authority issuing certificates. To exclude private keys from the adversary's knowledge, they are modeled as private free names, while public keys and certificates are defined using constructors and rewrite rules for issuing and verifying certificates. Additionally, we modeled rewrite rules to negotiate a session key and an algorithm providing authenticated encryption and decryption [34] based on symmetric cryptography. The model includes a setup process to establish preconditions, such as issuing the IA component's manufacturer certificate and configuring the CE with manufacturer and control domain CA public keys. Communication occurs over a public channel accessible to a Dolev-Yao adversary, capable of eavesdropping, modifying, or replaying messages. Taking possession and application credential provisioning are modeled in two ProVerif scripts using the same modeled functions and rewrite rules, differing in the behavior and configured credentials of the IA component and the CE.

For the unilateral authentication during taking possession, the IA component sends its manufacturer certificate and a signed ephemeral key exchange message with nonces to prevent replay attacks, derives a session key, and provides a proof of private key possession. The CE validates the certificate using the manufacturer CA's public key and verifies proof of possession, satisfying *req. 1*. Upon successful validation, an authenticated-encryption-protected channel using the negotiated session key is established, enabling the CE to provision a control domain credential, satisfying *req. 2* (see Fig. 10). Injective correspondence

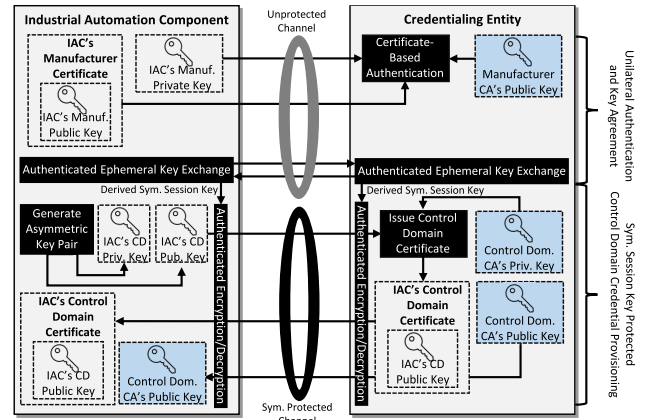


FIGURE 10. Taking possession and control domain (CD) credential provisioning sequence as modeled in ProVerif. The CE authenticates the IAC unilaterally, followed by an authenticated ephemeral key exchange and provisioning of a control domain credential via an authenticated-encryption-protected channel.

properties are used to formulate and prove the authenticity requirements.

For automated application credential provisioning, the IA component, now provisioned with a control domain credential, mutually authenticates with the CE using certificate-based authentication and negotiates a session key. Both entities exchange certificates, verify signatures, and check authorization attributes, satisfying *req. 3*. The IA component generates an application credential private key and certificate signing request (CSR), which the CE forwards to the CA. The issued application certificate and CA public key are provisioned via the authenticated-encryption-protected channel using the previously negotiated session key (see Fig. 7). We verify three security properties:

- *Private key secrecy:* Adversaries cannot derive private keys, verified via reachability queries.
- *Unilateral IA component authentication:* During taking possession, the CE authenticates the IA component, precluding rogue impersonation, verified by injective correspondence ensuring that acceptance by the CE always follows the IA component's initiation of the authentication process.
- *Mutual authentication:* In application credential provisioning, mutual authentication is verified via mutual injective correspondences.

All properties were successfully verified using ProVerif, confirming the model's security under an unbounded number of sessions. Full ProVerif scripts are available at [33] for reproducibility. This analysis provides the first formal guarantees for IIoT credential management.

In particular, robustness checks validate the model's resilience. Deliberately leaking CA or EE private keys falsifies authentication properties, confirming the necessity to keep them private. Using the authentication-only-protection variant instead of authenticated-encryption yields equivalent security properties.

TABLE 2. Credential management functions and mean execution time (MET).

Name	Request	Response	MET [ms]
GetStatus	-	StatusInfo	0.004
GetCertPath	-	X.509 Cert. Path	28.85
GenKeyPairAndCSR	Key-Pair Alg.	CSR	593.81
ImprintTrustAnchor	X.509 Cert.	ResponseCode	111.41
ImprintEECertPath	X.509 Cert. Path	ResponseCode	121.85
ImprintCRL	X.509 CRL	ResponseCode	68.64
RemoveArtifact	-	ResponseCode	10.72

IX. IMPLEMENTATION

This section presents a CANopen-FD-specific implementation of the generic credential management model, validating its applicability to resource-constrained IIoT systems. CANopen FD, as standardized by the CAN in Automation [3], does not natively incorporate security extensions for the management and use of credentials. We developed a protocol concept and a physical demonstrator that integrates the IA components, CEs, and CAs, employs manufacturer, control domain, and application credentials, and realizes the credentialing processes. Details on the implementation and performance evaluations of certificate-based authentication are available in separate publications [6], [7].

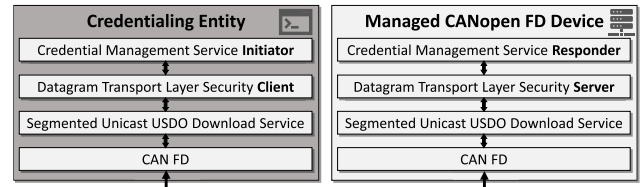
A. PROTOCOL CONCEPT AND MANAGEMENT FUNCTIONS

To realize the credentialing processes, a request/response communication between CEs (requesters) and managed CANopen FD components (responders) is developed, resulting in a push model communication paradigm. The CEs request IA components to execute functions and send subsequent responses. The functions can be grouped according to the triggered result:

- responding status information or a certification path already present on the device,
- the persistence of a credential object element which was either locally generated or provisioned by a CE, so-called *imprinting*, and
- the removal of a credential object.

Table 2 summarizes the functions and defines their payload. Each request contains the information to which credential type it refers to. Per credential type, there is one credential object instance to be managed and used. All imprinting and removal functions are either positively or negatively confirmed. The *StatusInfo* response is a one byte value indicating the presence or absence of a credential type. X.509 certificates according to the profile of IEEE 802.1AR [10] are used as certificates and trust anchors. The current implementation supports secp256r1 key pairs, with an option for simple extension to various algorithms [10].

Since CANopen FD does not incorporate security extensions that employ credentials and allow the establishment of protected communication channels (cf. Section VIII), we propose to use the Datagram Transport Layer Security (DTLS) protocol. It supports unilateral and mutual

**FIGURE 11.** The layers of the CANopen-FD-specific credential management protocol comprise CAN FD, CANopen FD, DTLS, and the credential management request/response communication.

certificate-based authentication using the DTLS Handshake protocol and an authenticated-encryption protection of the subsequent exchange of credential management requests and responses using the DTLS Application Data protocol. This matches the properties of the communication channel of our formal security analysis. The CE acts as the DTLS client, allowing it to actively initiate certificate management sessions, whereas managed devices act as DTLS servers. Additionally, CEs are co-located with the CANopen Network Management (NMT) entity allowing it to proceed credentialing processes prior to instructing devices to change into their operational state. The DTLS records are conveyed using CANopen FD's segmented unicast Universal Service Data Object (USDO) download protocol, capable of transmitting large payloads up to 4.29 GB [3]. The data objects are written to the device's object dictionary at a static offset in the implementation-specific profile area, allowing simultaneous transfers to multiple devices using different node IDs. Fig. 11 summarizes the layering of the used protocols.

For taking possession, the CE unilaterally authenticates the CANopen FD device during the DTLS handshake using an exemplary manufacturer credential, validated with the manufacturer's CA trust anchor, and verifies the device instance via the CANopen vendorID and serial number in the certificate's *subjectAlternativeName* field. A DTLS-protected channel is established post-handshake to provision control domain credentials. For provisioning and updating, the CE triggers the *GenKeyPairAndCSR*, *ImprintTrustAnchor*, *ImprintEECertPath*, and *ImprintCRL* functions. For decommissioning and abandoning possession, the *RemoveArtifact* function is used, followed up by an *ImprintCRL* with the updated CRL on the remaining devices.

B. PHYSICAL DEMONSTRATOR

The physical demonstrator that implements the protocol concept comprises three CANopen FD devices implementing the responder and two CEs implementing the requester (see Fig. 12). All requesters, deployed on NXP LPC55S16-EVK boards with a clock frequency of 150 MHz, and responders, deployed on Raspberry Pi 4Bs with a clock frequency of 1.5 GHz, use the Micro CANopen FD [35] and mbedTLS [36] protocol stack implementations.

Due to the generic model's TOFU characteristic, taking possession occurs over a 1:1 CAN FD connection using

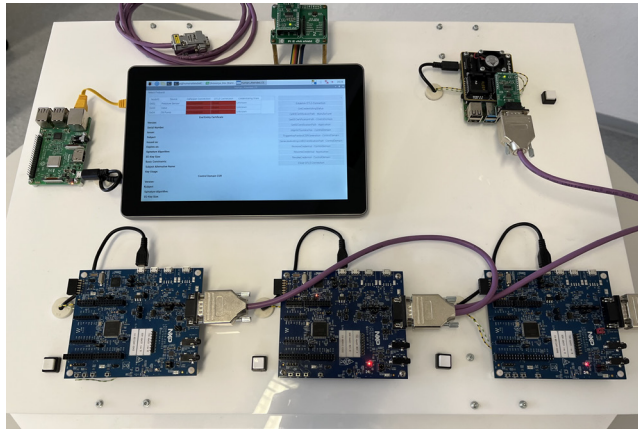
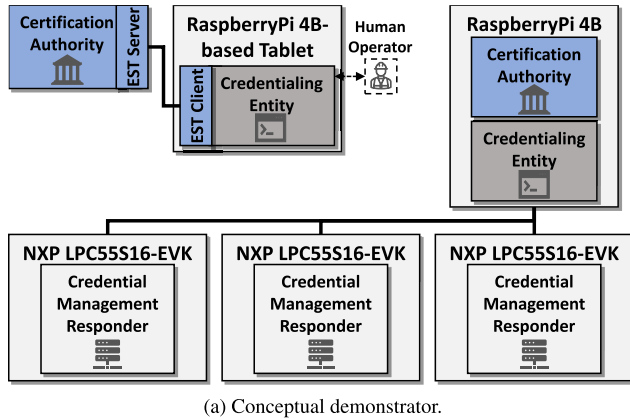


FIGURE 12. The credential management model is validated via a CANopen-FD-specific implementation and demonstrator.

a tablet-based and human-operated CE to prevent unauthorized access. Operators can inspect IA component's certification paths that were validated within the DTLS handshake and manually instruct IA components which credential management functions to perform. To showcase the external CA variant, the CE uses the Enrollment over Secure Transport (EST) protocol [37] for CA communication. EST provides built-in mechanisms to enable mutual authentication and protected communication between CAs and the CEs using the Transport Layer Security protocol (TLS) [38].

For automated application credential provisioning, performed by the autonomously acting CE, mutual authentication is achieved using the previously provisioned control domain credentials, with the CE's certificate containing a *subjectAlternativeName* extension to verify its management authorization. Additionally, the autonomously acting CE is capable of renewing credentials.

The implementation validates all processes: initial manufacturer credential-based taking possession, automated application credential provisioning, renewal via CSR generation and certificate imprinting, and off-boarding with credential removal and CRL updates. This demonstrates the model's feasibility for resource-constrained IIoT networks, enabling secure and scalable credential management.

Mean Execution Time of Credential Management Functions

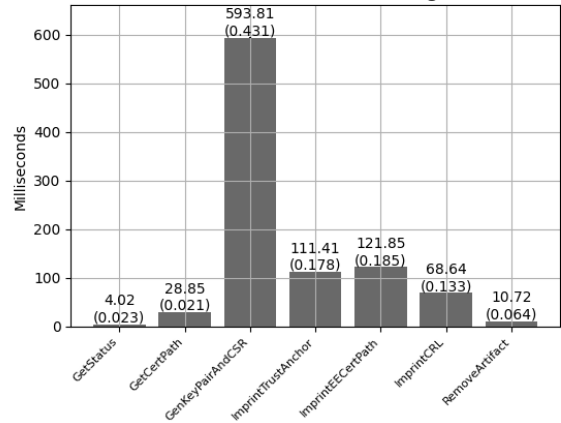


FIGURE 13. Experimental mean execution time and standard deviation results of credential management functions using the demonstrator implementation.

C. PERFORMANCE EVALUATION

Since it seems to be a prevalent assumption that asymmetric cryptographic algorithms impose an excessive computational load for resource-constrained devices and to validate the practical feasibility of our model, we performed a twofold analysis.

First, in a separate publication [7], we analyzed the performance of certificate-based authentication within TLS on an embedded platform [7]. The TLS protocol family belongs to the most widespread and most mature security protocols and is also used within PROFINET, CIP Security, and IEC/IEEE 60802. To this end, session establishments on an embedded ARM Cortex platform are analyzed. To be precise, the execution time to generate and parse TLS Handshake protocol messages for the client and server sides and in particular the impact of different algorithms for the ephemeral Diffie-Hellman key exchange and the impact of different lengths and subject public key algorithms of certification paths is studied.

Second, we analyze the runtime of credential management functions from the beginning of the request generation until receiving and parsing the response message. We limit ourselves to analyzing application credential management functions using the autonomously acting CE, as they impose a higher time criticality, e.g., during the replacement of a device. For the measurements, the autonomously acting CE and one IA component repeat each function for 100 iterations and we calculate the mean and standard deviation. The results of the mean execution time are given in Table 2 and Fig. 13. The variation between the 100 iterations was negligible.

The results clearly show that the *GenKeyPairAndCSR*, using asymmetric cryptography, is by far the most time consuming function with 593.8 ms on average. The *imprinting* functions, comprising write accesses to the file system and requiring the transportation of multiple hundreds of bytes, follow with 111.4 ms

(*ImprintTrustAnchor*), 121.85 ms (*ImprintEECertPath*), and 68.64 ms (*ImprintCRL*). Removing artifacts and responding status information are the least time consuming.

Both analyses confirm that the use of asymmetric cryptographic algorithms is indeed feasible on resource-constrained devices, allowing the use of the well-proven (D)TLS protocol and credential management functions also for applications from the IIoT including fieldbus communication.

X. CONCLUSION

This paper presents a generic credential management model for secure field-level communication in IIoT networks, addressing the lack of a unified, protocol-agnostic, and lifecycle spanning approach. The model's novel elements — a credentialing entity, dedicated control domain credentials, and lifecycle-oriented processes — enable secure, scalable, and protocol-agnostic credential management, satisfying security, functional, and usability requirements. A CANopen-FD-specific implementation on a physical demonstrator and detailed performance analysis of certificate-based authentication within security protocols validate the model's feasibility for resource-constrained systems. Formal verification addresses the lack of current standards to formally prove the security properties of the model confirming robustness against impersonation and unauthorized access.

Future work will focus on optimizing performance for real-time IIoT applications, developing lightweight security protocols, and extending the model to additional technologies like 5G-campus-networks.

REFERENCES

- [1] *Industrial Communication Networks—Fieldbus Specifications*, document IEC 61158, 2023.
- [2] *OPC 10000-12 UA Part 12, V1.05.02*, OPC Foundation, Scottsdale, AZ, USA, Nov. 2022.
- [3] *CANopen FD Application Layer and Communication Profile, CAN in Automation 1301*, CAN Automat. e.V., Nürnberg, Germany, Sep. 2017.
- [4] K. E. Hemsley and R. E. Fisher. (Dec. 2018). *History of Industrial Control System Cyber Incidents*. [Online]. Available: <https://www.osti.gov/biblio/1505628>
- [5] J. Göppert, A. Walz, and A. Sikora. (Apr. 2024). *A Survey on Life-Cycle-Oriented Certificate Management in Industrial Networking Environments*. [Online]. Available: <https://www.mdpi.com/2224-2708/13/2/26>
- [6] J. Göppert and A. Sikora, "Credential management for CANopen FD: A life cycle oriented concept and implementation," in *Proc. IEEE 22nd Int. Conf. Ind. Informat. (INDIN)*, Aug. 2024, pp. 1–6. [Online]. Available: <https://ieeexplore.ieee.org/document/10774305>
- [7] J. Göppert, A. Chomel, and A. Sikora, "Performance evaluation of TLS session establishments on an ARM cortex-M4 platform," in *Proc. IEEE 12th Int. Conf. Intell. Data Acquisition Adv. Comput. Systems: Technol. Appl. (IDAACS)*, Sep. 2023, pp. 1205–1210. [Online]. Available: <https://ieeexplore.ieee.org/document/10348947>
- [8] J. Göppert and A. Sikora, "Formal modeling and verification of generic credential management processes for industrial cyber-physical systems," *IEEE Embedded Syst. Lett.*, vol. 17, no. 5, pp. 349–352, Oct. 2025. [Online]. Available: <https://ieeexplore.ieee.org/document/11205915>
- [9] B. Blanchet. (2016). *Modeling and Verifying Security Protocols With the Applied Pi Calculus and ProVerif*. [Online]. Available: <http://www.nowpublishers.com/article/Details/SEC-004>
- [10] *Secure Device Identity*, IEEE Standard 802.1AR, Aug. 2018.
- [11] S. Boeyen, "Internet X.509 public key infrastructure certificate and CRL profile," Internet Engineering Task Force, Wilmington, NC, USA, Tech. Rep. IETF RFC 5280, May 2008. [Online]. Available: <https://datatracker.ietf.org/doc/rfc5280>
- [12] J. S. Rinaldi. (Jun. 2019). *CIP Certificate Management Object*. [Online]. Available: <https://www.rtautomation.com/rtas-blog/cip-certificate-management-object/>
- [13] *IEC/IEEE 60802 TSN Profile for Industrial Automation*. Accessed: Nov. 28, 2025. [Online]. Available: <https://1.ieee802.org/tsn/iec-ieee-60802/>
- [14] *OPC 10000-21 UA Part 21*, OPC Foundation, Scottsdale, AZ, USA, Nov. 2022.
- [15] M. Badra, A. Luchuk, and J. Schönwälder. (Jun. 2015). *Using the NETCONF Protocol Over Transport Layer Security (TLS) With Mutual X.509 Authentication*. [Online]. Available: <https://datatracker.ietf.org/doc/rfc7589>
- [16] M. Runde, S. Hausmann, C. Tebbe, B. Czybik, K.-H. Niemann, S. Heiss, and J. Jasperneite, "Schlussbericht SEC," Hochschule Hannover, Hannover, Germany, Tech. Rep. 812809483, Dec. 2014, doi: [10.25968/opus-499](https://doi.org/10.25968/opus-499).
- [17] G. Karthikeyan and S. Heiss, "PKI and user access rights management for OPC UA based applications," in *Proc. IEEE 23rd Int. Conf. Emerg. Technol. Factory Automat. (ETFA)*, Turin, Italy, Sep. 2018, doi: [10.1109/ETFA.2018.8502603](https://doi.org/10.1109/ETFA.2018.8502603).
- [18] F. Kohnhäuser, D. Meier, F. Patzer, and S. Finster, "On the security of IIoT deployments: An investigation of secure provisioning solutions for OPC UA," *IEEE Access*, vol. 9, pp. 99299–99311, 2021. [Online]. Available: <https://ieeexplore.ieee.org/document/9478911>
- [19] A. A. Krishnan, S. K. Rajendran, and T. K. Sunil Kumar, "Improved PKI certificate lifecycle management with centralized device management for industrial IoT," in *Proc. IEEE Int. Conf. Public Key Infrastructure Appl. (PKIA)*, Sep. 2022, pp. 1–5. [Online]. Available: <https://ieeexplore.ieee.org/document/9952216>
- [20] C.-S. Park and H.-M. Nam, "Security architecture and protocols for secure MQTT-SN," *IEEE Access*, vol. 8, pp. 226422–226436, 2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9296847>
- [21] V. Danilchenko, M. Theobald, and D. Cohen, "Bootstrapping security configuration for IoT devices on networks with TLS inspection," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2019, pp. 1–7. [Online]. Available: <https://ieeexplore.ieee.org/document/9024325>
- [22] J. Höglund, S. Lindemer, M. Furuheid, and S. Raza, "PKI4IoT: Towards public key infrastructure for the Internet of Things," *Comput. Secur.*, vol. 89, Feb. 2020, Art. no. 101658. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404819302019>
- [23] J. Astorga, M. Barcelo, A. Urbietta, and E. Jacob, "How to survive identity management in the industry 4.0 era," *IEEE Access*, vol. 9, pp. 93137–93151, 2021. [Online]. Available: <https://ieeexplore.ieee.org/document/9464249>
- [24] J. Höglund, M. Furuheid, and S. Raza, "Lightweight certificate revocation for low-power IoT with end-to-end security," *J. Inf. Secur. Appl.*, vol. 73, Mar. 2023, Art. no. 103424. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2214212623000091>
- [25] C. Boudagdigue, A. Benslimane, and A. Kobbane, "Cluster-based certificate revocation in industrial IoT networks using signaling game," in *Proc. IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–6. [Online]. Available: <https://ieeexplore.ieee.org/document/93224971>
- [26] A. Garba, D. Khoury, P. Balian, S. Haddad, J. Sayah, Z. Chen, Z. Guan, H. Hamdan, J. Charafeddine, and K. Al-Mutib, "LightCert4IoTs: Blockchain-based lightweight certificates authentication for IoT applications," *IEEE Access*, vol. 11, pp. 28370–28383, 2023. [Online]. Available: <https://ieeexplore.ieee.org/document/10076434/>
- [27] (2018). *RAMI 4.0 Platform Industrie 4.0*. [Online]. Available: <https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/rami40-eine-einfuehrung.pdf?blob=publicationFile&v=7>
- [28] S. Hausmann and S. Heiss, "Usage of public key infrastructures in automation networks," in *Proc. IEEE 17th Int. Conf. Emerg. Technol. Factory Automat. (ETFA)*, Krakow, Poland, Sep. 2012, doi: [10.1109/ETFA.2012.6489750](https://doi.org/10.1109/ETFA.2012.6489750).
- [29] F. Stajano. (2000). *The Resurrecting Duckling—What Next?*. [Online]. Available: <https://www.cl.cam.ac.uk/~fms27/papers/2001-Stajano-duckling.pdf>
- [30] *Power Systems Management and Associated Information Exchange—Data and Communications Security*, document IEC 62351-9, 2017.
- [31] S. Santesson. (Jun. 2013). *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol—OCSP*. [Online]. Available: <https://datatracker.ietf.org/doc/rfc6960>

- [32] Y. N. Pettersen, *The Transport Layer Security (TLS) Multiple Certificate Status Request Extension*, document RFC 6961, Jun. 2013. [Online]. Available: <https://datatracker.ietf.org/doc/rfc6961>
- [33] (May 2025). *Repository*. [Online]. Available: <https://anonymous.4open.science/r/Formal-Credential-Management-Model-8EBC/README.md>
- [34] T. Mieno. (Jul. 2023). *Formal Verification of Authenticated Encryption With Associated Data With Tamarin Prover*. [Online]. Available: <https://ieeexplore.ieee.org/document/10487609/>
- [35] (Sep. 2023). *MicroCANopen, Embedded Systems Academy*. [Online]. Available: <https://canopenstore.com/products/microcanopen-plus>
- [36] (Nov. 2025). *MbedTLS: An Open Source, Portable, Easy to Use, Readable and Flexible TLS Library, and Reference Implementation of the PSA Cryptography API*. [Online]. Available: <https://github.com/Mbed-TLS/mbedtls>
- [37] M. Pritikin, P. E. Yee, and D. Harkins, *Enrollment Over Secure Transport*, document RFC 7030, Oct. 2013. [Online]. Available: <https://datatracker.ietf.org/doc/rfc7030>
- [38] E. Rescorla. (Aug. 2018). *The Transport Layer Security (TLS) Protocol Version 1.3*. [Online]. Available: <https://datatracker.ietf.org/doc/rfc8446>



JULIAN GÖPPERT received the B.Eng. degree in mechanical engineering from Baden-Württemberg Cooperative State University, Karlsruhe, in 2019, and the M.Sc. degree in automotive systems engineering from Karlsruhe University of Applied Sciences, in 2021. He is currently pursuing the Ph.D. degree with the Institute of Reliable Embedded Systems and Communication Electronics (ivESK), Offenburg University of Applied Sciences, in the field of security in industrial communication. He is the author and co-author of several research articles. He is an Active Member of the Security Working Groups of PROFIBUS and PROFINET International (PI) and the Interest Group Safety and Security of the CAN in Automation (CiA).



ANDREAS WALZ received the Dipl.-Phys. degree in physics from the University of Freiburg and the Ph.D. degree in testing approaches for implementations of secure communication protocols from the University of Freiburg, in 2025. Then, he joined ivESK, Offenburg University of Applied Sciences, as a Research Engineer. After joining the automation industry, he continues working in the field of secure communications for embedded devices, with a particular focus on industrial automation systems. He is the author and co-author of several research articles. He is an Active Member of the Security Working Groups of PROFIBUS and PROFINET International (PI) and the Industrial Ethernet Security Harmonization Group (IESHG).



AXEL SIKORA received the Dipl.-Ing. (corresponding to M.Sc.) and Dipl.-Wirt.-Ing. (corresponding to M.B.A.) degrees from RWTH Aachen, in 1995 and 1997, respectively. His doctoral thesis was on the design of high-temperature circuits using SOI technologies with Fraunhofer IMS, Duisburg. After several positions in telecom and semiconductor industry, he joined Baden-Württemberg Cooperative State University, in 1999, and Offenburg University, in 2011, as an Associate Professor, where he is currently the Scientific Director of ivESK. From 2016 to 2025, he was a Deputy Member of the Board with the Hahn-Schickard Institute. He has published more than 300 scientific articles. He is the author, co-author, and an editor of several textbooks, issued 12 patents, and founded and co-founded five startup companies. He is a member of manifold international program committees.

• • •