Remote Attestation for Secure Industrial Device Onboarding

Volodymyr Trykoz[1]✉, Björn Leander[2], Saad Mubeen[1], and Mohammad Ashjaei[1]

[1] Mälardalen University, Västerås, Sweden
[2] ABB Process Automation, Process Control Platform, Västerås, Sweden
firstname.lastname@mdu.se[1]; firstname.lastname@se.abb.com[2]

Abstract. Security in industrial automation systems becomes more complicated as industry moves toward modern and flexible solutions such as modular automation and increased vendor diversity. Attack surfaces grow as more network interfaces are added and more devices become exposed to the internet and cloud. The rapid introduction and replacement of devices make it difficult to track vulnerabilities and ensure nothing malicious enters the system. Hence, trusting a device prior to its onboarding becomes critical for ensuring the overall security and safety of the system. This paper proposes a remote attestation mechanism for the onboarding process which follows the Remote ATtestation ProcedureS proposals. In order to provide evidence for the applicability of the proposed process, we provide a reference implementation. Moreover, we provide a security analysis of the proposed solution in the form of a threat model.

1 Introduction

As Industry 4.0 [3] becomes more and more mainstream, complex industrial systems are required to be flexible, scalable, and adaptable. This brings new challenges from the security perspective, as more attack surfaces open up due to support for flexibility and adaptability. The question of trust becomes imperative for secure operations of industrial automation systems, many of which may operate in safety-critical environments [4]. The systems must support flexibility by allowing users to quickly introduce new entities without compromising security. Without disrupting essential functionality, the system must be able to onboard a new device, provided it is deemed trustworthy. Modular automation [1] is an automation system design strategy allowing such dynamic behavior to support, e.g., scaling without interrupting operations.

In industrial automation systems, the Open Platform Communications Unified Architecture (OPC UA) is a widely used communication protocol in industrial automation and Industry 4.0 [21]. It has extensive security considerations and a part dedicated to secure onboarding of automation devices [10]. However, the OPC UA onboarding process focuses on verifying identity, and does not provide any mechanism for validating device state. This limits the extent to which

the onboarded device can be trusted. Even when the identity of the device is verified, its integrity cannot be fully assured. Consequently, it may be compromised by malware or firmware with critical vulnerabilities.

In the context of flexible industrial automation systems, remote attestation can be used for online state verification. During remote attestation, a device collects measurements about its software and configuration, and sends this evidence to a remote entity that can appraise it. Based on these measurements, the verifier determines whether the device is running permitted software or if deviations are present. The measurements are collected in a way that makes tampering extremely difficult without compromising the hardware root of trust, and they are cryptographically protected to ensure authenticity and integrity. In this regard, the Remote ATtestation ProcedureS (RATS) [5] working group defines a vendor-independent architecture, interfaces, and message types to enable secure remote attestation and related operations. It standardizes key actors and various processes used for attestation flows [7].

Previous works on integrating remote attestation with OPC UA focuses on isolated attestation and do not consider cross-protocol scenarios or the onboarding process [8, 12]. In this paper, we provide a comprehensive OPC UA mechanism for performing remote attestation during secure onboarding, which hardens the system against compromised devices. The paper provides the following contributions.

1. It proposes an attestation-enhanced OPC UA onboarding process using the models and workflows proposed by RATS.
2. It presents a reference implementation to provide evidence on how this integration can be performed.
3. It provides a security analysis, based on threat modeling using the STRIDE classification, that identifies and evaluates relevant threats, upon which we subsequently propose mitigation strategies.

2 Related Work

There are very few works to date that have focused on integrating remote attestation into OPC UA. Birnstill *et al.* [8] investigate a basic model for retrieving Trusted Platform Module (TPM) quotes through OPC UA and verifying them. Their approach focuses exclusively on evidence transmission, and does not consider the broader onboarding workflow.

Another OPC UA–related attestation effort is presented by Juhola *et al.* [12]. They propose a simple reference attestation sequence in which a server provides evidence in response to a nonce-based request. The server aggregates evidence by interacting with its subsystems. Similar to [8], their work does not position attestation within the OPC UA onboarding process. They also do not adopt the RATS approaches and workflows, although they mention it as future work.

Matischek *et al.* [16] propose integrating a hardware-based secure element into OPC UA to offload certain cryptographic operations. Although their work

does not address onboarding or remote attestation directly, it demonstrates how secure elements serve as essential building blocks for strong attestation guarantees, highlighting a complementary direction for strengthening OPC UA device trustworthiness.

With respect to reference implementations based on the RATS architecture, Usman and Asplund [20] provide a prototype implementation that integrates the RATS actors. Although the overall direction aligns with ours, their work does not address the OPC UA onboarding model, which is the focus of this paper.

In the context of OPC UA onboarding, Kohnhäuser *et al.* [14], later expanded in [17], analyze secure onboarding using DevIDs and hardware security modules. They focus on identity verification by leveraging DevIDs and associated cryptographic hardware. Our work builds further by assuming the identity check has already succeeded and extending the onboarding workflow with state verification via remote attestation.

3 Remote Attestation Process Proposal

The proposed onboarding model builds on two elements of trustworthiness: identity and state. To present the solution, we first describe the trust model and show how the current OPC UA onboarding process fits into it.

3.1 Secure Onboarding and Device Trustworthiness

Before a new device is introduced into an operational environment, its *trustworthiness* must first be established. In this work, two aspects of trust establishment is considered:

- *Identity* — the device must provide credentials for its identity. This authenticates the device, and protects against man in the middle attacks.
- *State* — the software on the device must be in an updated and uncompromised condition. This includes confirming that its software stack has not been tampered with (e.g., via rootkits), it is not running vulnerable or outdated components, and updates have been applied successfully.

When both aspects can be verified, the device is considered sufficiently *trustworthy* to join the system. At this point, the device can be provisioned with the necessary operational credentials, such as certificates, and with information regarding which entities it should trust. The onboarding workflow should be as streamlined as possible, with identity and state verification automated, remotely executable, and requiring minimal physical interaction with the device.

Using Device Identifiers (DevIDs) [11] is one mechanism for device identity verification. They are manufacturer-issued credentials that uniquely identify a specific device. A device owner or operator can authenticate these credentials by obtaining the corresponding verification material from the original manufacturer and comparing them to the credentials embedded inside the device. The specific validation procedures are outside the scope of this work; however, introducing

the DevID concept is necessary for establishing the broader context in which our approach operates.

Remote attestation can be used for state verification, which is the process of verifying the integrity of a device or system (referred to as the *attester*) from a remote location. The attester generates *measurements* of its configuration (such as its boot software stack or other security-relevant components) and submits these measurements to a verifying entity known as the *verifier* [2]. The collected measurements constitute *evidence* that the system is in a specific state. The verifier compares the received evidence to a set of trusted reference values in order to determine whether the attester's configuration has been altered. Two common models exist for remote attestation: the *background-check* model, shown in Fig. 1(a) and the *passport* model, illustrated in Fig. 1(b) [18].

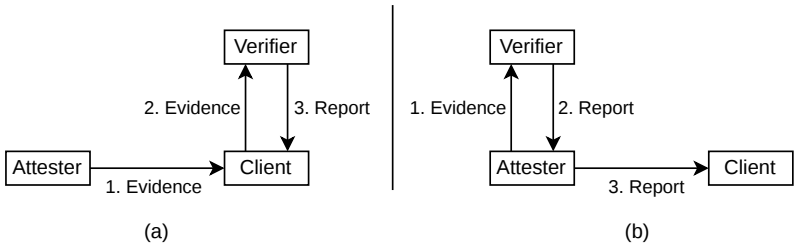

Fig. 1. Background check (a) and passport (b) models of remote attestation

In the background-check model, the attester sends measurements to the client (relying party), which forwards them to the verifier. The client relies on the attestation results to decide how to proceed. The verifier evaluates the evidence, produces an attestation result, and returns it to the client. In the passport model, the attester sends its measurements directly to the verifier and forwards the result to the client.

3.2 Aligning the OPC UA Onboarding Workflow with the Device Trustworthiness Model

OPC UA Part 21 [10] is focused on describing a secure onboarding model for new devices. Global Discovery Server (GDS) is one of the core entities in this process. Among its many roles, the GDS functions as a certificate authority, providing CA-signed certificates and trust lists to support PKI-based trust.

The current OPC UA onboarding process provides device authenticity, but lacks state validation. This means that although we can be reasonably confident that we are onboarding the correct device, we cannot say anything about the software or firmware it is running, whether it has been altered or corrupted, or whether it is running versions with known vulnerabilities. As a result, we risk provisioning a compromised device, which opens up a new attack surface for

adversaries attempting to get entry into the system. Therefore, it does not fulfill the trust establishment requirements described in previous sections.

To address this limitation, we extend the procedure with a RATS-based attestation workflow to improve the trustworthiness assessment. The integration of the RATS interaction model into the onboarding process is illustrated in Fig. 2. The original steps remain unchanged, but a state-verification step is inserted into the middle of the workflow.

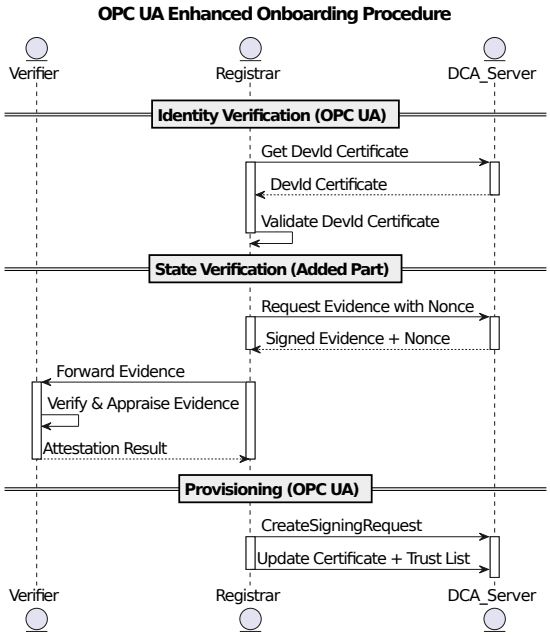

Fig. 2. Sequence diagram for the enhanced OPC UA onboarding model. The enhanced part is marked, and requires a new entity, the *verifier*, to be part of the onboarding.

The procedure for *state* verification is based on the RATS reference interaction models for remote attestation [7]. The registrar requests evidence that can be used to asses the state of the device. It sends a nonce that must be embedded into the evidence to prevent replay attacks. The device incorporates the nonce into the evidence, signs it, and returns it to the registrar. The registrar forwards the evidence to a verifier, which cryptographically validates its legitimacy by checking the signature and the included nonce. If the evidence is cryptographically valid, the verifier proceeds with appraisal and produces a report summarizing the device's state. This report is sent back to the registrar, which performs its own appraisal and decides whether to continue with provisioning. If the results indicate the device is in a corrupted or unhealthy state, the onboarding process is aborted. If the appraisal is successful, provisioning proceeds.

This workflow allows verification of both identity and state aspects of the trustworthiness, ensuring that only trusted devices are provisioned.

3.3 Interaction Model for Attested Onboarding

In order to make the interaction work, several adjustments are necessary to the core OPC UA entities in terms of their responsibilities and interactions. Fig. 3 presents the high-level attestation system architecture, showing the involved entities and their interactions. The interactions are partially based on RATS interaction models and partially on the OPC UA onboarding model. The background-check model is used, as it fits the OPC UA worfklow, where there is an administrative client (Registrar) responsible for verifying and provisioning a device. The proposed approach naturally extends the client-centric workflow to also request and verify device evidence via the verifier.

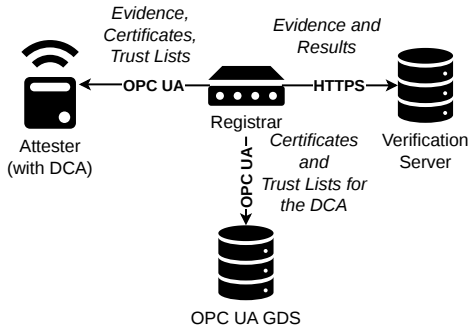

Fig. 3. Onboarding interaction model.

The following entities are part of the model.

- **Registrar** – initiates all interactions and acts as the central coordinator, requesting attestation evidence, forwarding it to the Verifier, and evaluating the result to ensure only devices with acceptable evidence receive operational certificates and trust lists. It maps to the Relying Party role in RATS.
- **Attester (DCA Server)** – generates verifiable evidence demonstrating its runtime integrity and allows the Registrar to update its operational certificate and trust list upon successful verification. It maps to the Attester role in RATS.
- **Verification Server** – evaluates evidence against device-specific appraisal policies and provides a verdict to assist the Registrar in determining whether the DCA Server can be safely provisioned. It maps to the Verifier role in RATS.
- **OPC UA GDS** – **G**lobal **D**iscovery **S**erver acts as the Certificate Authority (CA). It issues operational certificates and distributes trust lists post-attestation, enabling approved devices to join the network.

4 Reference Implementation

The reference implementation demonstrates the enhanced onboarding workflow. It is written in C# and available on GitHub[3]. Fig. 3 shows the system architecture. Since our work targets OPC UA, all interactions between the Registrar, Attester, and GDS use OPC UA channels to align with the onboarding model [10].

To demonstrate a multi-protocol scenario, the Registrar communicates with the Verification Server over HTTPS. This highlights how the Registrar can interact with a Verifier that exposes an HTTPS endpoint for submitting evidence and receiving appraisal results. The OPC UA Foundation .NET stack is used for the OPC UA components, while ASP.NET is used for the HTTPS endpoint.

Conceptual Message Wrappers (CMWs [6]) transport attestation evidence and results in a protocol-agnostic format. Each CMW includes a media type identifier (e.g., MIME type or CoAP [19] Format), enabling interoperable decoding across heterogeneous transport protocols.

The primary objectives of the implementation are to:

- Validate end-to-end evidence generation, encapsulation, transport across heterogeneous protocols (OPC UA and HTTPS), and independent verification.
- Demonstrate integration of the attestation into the OPC UA push management onboarding workflow without altering the specification.
- Providing a minimal yet extensible foundation for future experimentation with alternative evidence formats, advanced attestation concepts, and more complex onboarding scenarios.

The prototype supports two different boot configurations for the attester. One configuration matches the reference values stored on the verification server, while the other intentionally deviates from them. This allows to demonstrate both a successful attestation flow and an unsuccessful one.

5 Threat Analysis

We want to ensure that our system does not expose any vulnerabilities that could allow attestation to be bypassed. We also want the system to remain available, avoid disclosing unnecessary information about itself, and maintain proper visibility into its own operations. We therefore perform threat modeling using the STRIDE [9] approach. STRIDE categorizes threats as **S**poofing, **T**ampering, **R**epudiation, **I**nformation Disclosure, **D**enial of Service, and **E**levation of Privilege. This approach enables a systematic security analysis of the system from the perspective of each threat category.

The methodology is inspired by [13, 15] and summarized in Fig. 4. First the threat consequences are identified, then the system's assets and their trust boundaries are enumerated, followed by identification of the attack surfaces through which an attacker may attempt to achieve its goals. Each attack surface is analyzed according to STRIDE, and possible mitigations are outlined.

[3] https://github.com/vovatrykoz/UnifiedAttestation

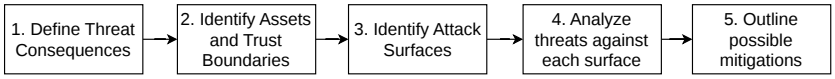

Fig. 4. The threat analysis methodology

5.1 Threat Consequences

Each threat consequence is related to the attestation-enhanced onboarding workflow and may have severity on the scale: *Low*, *Moderate*, *High*, and *Severe*.

TC1 A device is onboarded while bypassing attestation. ⇒ *Severe*
TC2 Denial-of-Service (DoS) attack prevents a legitimate device from being onboarded. Threat to the availability of the device. ⇒ *Moderate*
TC3 DoS attack prevents the attestation system from performing its role. Threat to the availability of the onboarding system. ⇒ *High*
TC4 Attestation evidence or appraisal results are revealed to unauthorized parties. Threat to the confidentiality of attestation information. ⇒ *Low-Moderate*
TC5 The system denies having performed actions that actually happened or claims to have performed actions that never took place. Threat to non-repudiation of the system. ⇒ *Low-Moderate*

5.2 Assets and Trust Boundaries

The system assets along with their associated data-flows and trust boundaries are illustrated in Fig. 5. All the network connections and the Attester are seen as potential attack surfaces. We assume certain entities are already securely onboarded and trusted. Potential compromises of these entities are out of scope, as this work focuses on device onboarding.

- **P1: Attester**. Device to be onboarded. It interacts only with the Registrar over **DF1**.
- **P2: Registrar**. The Registrar is assumed to be uncompromised and trusted to perform its role as the attestation orchestrator. It therefore lies within a trust boundary. It initiates all interactions, and all other assets interact with it in response to its requests.
- **P3: Verification Server** is assumed to correctly perform evidence appraisal; therefore, it lies within a trust boundary. It interacts with the Registrar over data flow **DF2**
- **P4: OPC UA Global Discovery Server (GDS)**. It is involved in the provisioning step and plays a role in several threats related to bypassing the attestation workflow. It interacts with the Registrar (**DF3**).

5.3 Threats and mitigations enumeration

Threats are enumerated based on the STRIDE classification model, considering the defined attack surfaces, i.e., **P1** as well data-flows **DF1**–**3**. Additionally mitigations (M) are considered for each of the threats. The resulting threat analysis is summarized in Table 1, excluding non-applicable categories.

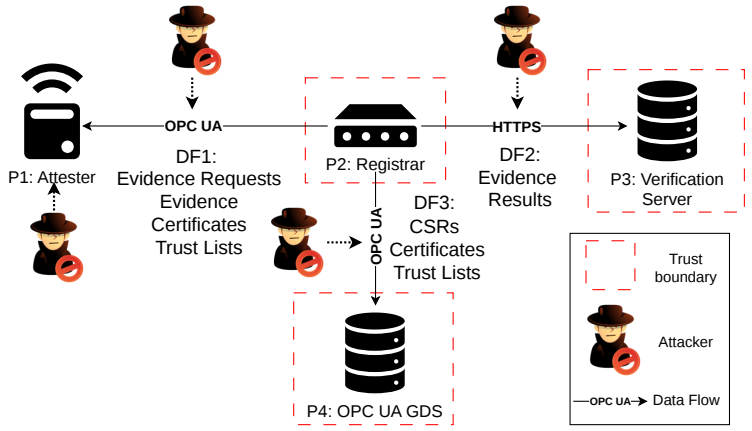

Fig. 5. Interaction diagram showing data flows and entities that can be manipulated by attackers under our threat model

Attester attack surface (P1). An attacker may try to spoof a device's identity to appear authentic and be onboarded (Threat 1), leading to **TC1**. To mitigate this, identity verification must occur before attestation (M1). In OPC UA, the registrar verifies DevID certificates against reference credentials and rejects mismatches [10]. Because DevID private keys are stored in hardware security modules (e.g., TPMs), forging an identity is infeasible unless an attacker physically steals a genuine device. Requiring attestation evidence to be signed with a dedicated attestation key (M2) further increases the difficulty of compromising the device.

If an attacker obtains a legitimate device provisioned with an DevID and attestation keys, they may attempt to submit falsified evidence that does not reflect the software actually running on the device (Threat 2), again targeting **TC1**. This is a critical threat: an attacker in control of the device could manipulate the attestation process to measure or report only uncompromised components while omitting compromised ones, making the evidence appear healthy despite a compromise. If the attacker accesses the device's signing credentials, they can generate evidence that is cryptographically valid but still misleading.

A hardware security module (HSM) (M3), such as a TPM, mitigates the risk of an attacker altering measurements, tampering with the attestation process, or accessing device credentials, thereby also strengthening M1 and M2. Additional hardening can be achieved through secure boot (M4) and runtime measurements (M5). Secure boot prevents unauthorized components from executing during startup, while runtime measurements monitor the device after boot to detect later modifications. A residual risk remains that a highly skilled attacker could bypass these protections, but doing so would require significant time and expertise and is likely to trigger at least one of the defenses.

Table 1. Threat analysis overview

Surface	Category	ID	Description	Consequences	Mitigations
P1: Attester	S	1	Attacker spoofs a legitimate device	TC1	M1, M2
	T	2	Attacker sends false evidence	TC1	M3, M4, M5
DF1: Registrar to Attester	S	3	Attacker spoofs the registrar	TC4	M6, M7, M8
	S	4	Attacker spoofs the attester	TC1	M1, M2, M9
	T	5	Attacker falsifies evidence	TC1, TC2	M2, M9
	T	6	Attacker replays older evidence	TC1, TC2	M10
	T	7	Attacker appends their certificate to a legitimate trust list	TC1, TC2	M9
	T	8	Attacker performs a relay attack	TC1	M1, M9
	R	9	Registrar fails to log events	TC5	M11
	I	10	Attacker intercepts evidence	TC4	M9
	D	11	Network-level DoS via packet dropping or oversized evidence payloads	TC2, TC3	M12, M13
DF2: Registrar to Verifier	S	12	Attacker spoofs the verifier	TC1, TC2	M9
	S	13	Attacker spoofs the registrar	TC4	M14
	T	14	Attacker falsifies evidence	TC1, TC2	M2, M9
	T	15	Attacker replays older evidence	TC1, TC2	M10
	T	16	Attacker replaces actual results with falsified results	TC1, TC2	M9, M15
	T	17	Attacker replays older results	TC1, TC2	M16
	R	18	Registrar or verifier fails to log events	TC15	M11
	I	19	Attacker intercepts evidence or results	TC6	M9
	D	20	Network-level DoS via packet dropping or oversized evidence/result payloads	TC2, TC3	M12, M13
	E	21	Unauthorized device sends evidence for verification	TC1, TC4	M14
DF3: Registrar to GDS	S	22	Attacker spoofs the registrar	TC1	M9
	S	23	GDS is spoofed	TC1	M9
	T	24	Attacker attempts to append their certificate to a legitimate trust list	TC1	M9
	R	25	Registrar or GDS fails to log events	TC5	M11
	I	26	Attacker intercepts certificates or trust lists	TC4	M9
	D	27	Network-level DoS via packet dropping, request flooding, or oversized payloads	TC2, TC3	M12, M13
	E	28	Unauthorized device requests certificates and trust lists from GDS	TC1	M14

Attester–Registrar attack surface (DF1). An attacker may attempt to spoof the Registrar to request attestation data from the Attester (Threat 3), targeting **TC4** to learn more about the Attester/attestation process. This is mitigated by having the Attester keep the attestation endpoint hidden until it receives a preliminary trust list. After verifying the Attester's identity, the Registrar provides a limited trust list (M6), e.g., containing only the Registrar's certificate. The Attester then exposes its attestation endpoint but interacts only with entities included in this list. A residual risk remains that an attacker may discover and attempt to follow this sequence. However, doing so would also exclude the attacker from onboarding, which is a more severe consequence, while obtaining any attestation data would pose only a low to moderate risk, as it typically consists of hashes that require external metadata to be meaningful.

A more serious threat from a spoofed Registrar is that an attacker could attempt to inject its own certificate into the device's trust list to gain backdoor

access. This can be mitigated by forcing a connection restart (M7) whenever the trust list is updated, making it practically infeasible for an attacker to capture the legitimate list, append its own certificate, and push an updated version in time. Limiting the Attester to a single simultaneous connection during onboarding (M8) further prevents such manipulation. Once onboarding is complete, the device trusts only legitimate system entities, reducing the attacker's opportunities. A residual risk remains that an attacker could provision a device by connecting first, but it would still lack access unless it also obtained a legitimate trust list from a genuine GDS and inserted its own certificate. Mitigations for this case are discussed in the DF3 section.

The reverse is also possible: an attacker may spoof the Attester to gain illegitimate access to the system (Threat 4), targeting **TC1**. This is mitigated through identity verification (M1) and the use of a secure, authenticated channel that ensures message-sender authenticity (M9). Additionally, attestation evidence should be signed using an attestation key (M2) that preferably resides in a HSM, providing an additional layer of identity protection. Combined, these measures strengthen identity trustworthiness. The residual risk is low, as an attacker would need to break the OPC UA security stack or compromise a TPM-protected key, which is considered practically infeasible.

An attacker may also replace legitimate evidence with falsified evidence in transit (Threat 5) or intercept and replay previously transmitted evidence (Threat 6). This may lead to **TC1** if corrupted evidence is replaced with healthy evidence, or to **TC2** if healthy evidence is replaced with corrupted-looking evidence. To prevent this, the evidence must be signed (M2), and the signed portion must include a nonce (M10). Signing ensures the evidence cannot be tampered with, while nonces prevent replay. Combined with M1, these measures assure that evidence cannot be forged over in transit over the network without physical access to a legitimate device.

An attacker may attempt to append their own certificate to a trust list in transit (Threat 7), achieving **TC1**. This is mitigated by using secure authenticated channels (M9), which provide message integrity and make tampering difficult. The residual risk considerations are similar to Threats 4 and 5.

An attacker may also perform a relay attack on the communication link (Threat 8), potentially inserting themselves into the system and bypassing attestation (**TC1**). DevID certificates help mitigate this (M1) by preventing unauthorized entities from initiating communication with the Registrar. However, a relay may still be possible through advanced network manipulation, allowing the attacker to insert themselves into an existing channel. This risk is further mitigated by using OPC UA secure authenticated channels (M9), which protect against meaningful message manipulation, such as tampering or replay, even if the attacker succeeds in relaying traffic. The residual risk is low and would require exploitable vulnerabilities in the OPC UA security stack implementation, which we consider unlikely.

If the Registrar does not keep logs of connections, such as device addresses and timestamps, troubleshooting becomes difficult (Threat 9), leading to **TC5**.

This is easily mitigated by requiring core entities to maintain logs and, if needed, forward them to a centralized auditing system (M11). With the security protections provided by OPC UA, attacks on such logging systems become infeasible.

If the channel is unprotected, an attacker may intercept evidence in transit without spoofing the Registrar (Threat 10), resulting in unauthorized information disclosure (**TC3**). Using secure authenticated channels (M9) prevents this, as encryption ensures that message contents cannot be read by an attacker. As before, the residual risk is minimal and would require breaking the OPC UA security stack.

An attacker may drop packets to disrupt onboarding or inject oversized payloads to crash the Registrar or Attester (Threat 11), targeting **TC2**. Mitigations include message size limits (M12) and network hardening (e.g., firewalls, M13), which reduce the impact of DoS attacks, though some residual overload risk remains.

Registrar–Verifier attack surface (DF2). A major threat on this link is an attacker spoofing a Verifier, allowing them to approve or reject arbitrary evidence and thereby permit or block devices from joining the network (Threat 12), achieving **TC1** or **TC2**. This is mitigated by using secure authenticated channels (M9): an HTTPS channel enforces client-side identity verification, making spoofing highly difficult. The residual risk is low and would require exploiting a vulnerability in the underlying Transport Layer Security (TLS) implementation.

The reverse is also possible: an attacker may spoof the Registrar and submit arbitrary evidence to learn about the Verifier's behavior or reverse engineer its verification logic (Threat 13), targeting **TC4**. Because API servers may not always authenticate clients, this can be mitigated by introducing an access-control mechanism (M14). The server would rely on tokens issued by an authorization service and grant or deny API access based on those tokens, which shifts authentication responsibility to an authorization entity. The residual risk depends on the security of the access-control system. Since this is a separate component, it is considered out of scope for this threat analysis.

Since the Registrar forwards attestation evidence to the Verifier, an attacker may replace valid evidence with falsified evidence (Threat 14) or replay older evidence (Threat 15), similar to DF1. These threats mirror Threats 5 and 6, so the same mitigations apply: signing (M2), secure authenticated channels (M9), and nonces (M10), along with the same residual-risk considerations.

An attacker may likewise replace (Threat 16) or replay (Threat 17) verification results on their return path to the Registrar, substituting negative results with positive ones to achieve **TC1**, or positive results with corrupted-looking ones to achieve **TC2**. As before, secure authenticated channels (M9) can protect the communication, while the results themselves should be signed (M15) and include a timestamp that defines their validity period (M16). This is analogous to Threats 5 and 6, with timestamps used instead of nonces. When properly implemented, and potentially combined with M1, these measures make manipulation infeasible unless the Verifier's security configuration is poor or the attacker gains physical access to it.

Troubleshooting this link is hindered if the Verifier does not keep logs of interactions and requests (Threat 18), leading to **TC5**. This is mitigated by M11, with residual-risk considerations similar to Threat 9.

An attacker may also intercept evidence or results in transit (Threat 19), leading to information disclosure and achieving **TC4**. This is mitigated by M9, with residual-risk considerations matching those of Threat 10.

A DoS attack may be launched by dropping packets or injecting oversized payloads to overwhelm either the Registrar or the Verifier (Threat 20), targeting **TC2**. Mitigations M12 and M13 apply here as well, with the same residual-risk considerations as in Threat 12.

Finally, an elevation of privilege may occur if an entity other than the Registrar communicates with the Verifier (Threat 21), potentially leading to **TC4**. In this case, the entity is not external but a previously onboarded, unauthorized one. This is mitigated by M14, with residual-risk considerations analogous to those discussed in Threat 13.

Registrar–GDS attack surface (DF3). An attacker on this link may attempt to spoof the Registrar and request certificates directly from the GDS, thereby bypassing attestation and obtaining credentials for a compromised device (Threat 22), achieving **TC1**. This is mitigated by using secure authenticated channels (M9), as the client is required to present a valid certificate, making Registrar spoofing difficult. The residual-risk considerations are similar to those in Threat 4.

A reverse scenario is also possible, where an attacker spoofs the GDS and performs its role by sending trust lists that include attacker-controlled certificates (Threat 23). This would grant the attacker backdoor access to the attester post-attestation, effectively allowing them to bypass attestation (**TC1**). This is mitigated by using secure authenticated channels (M9), as the Registrar must verify the GDS's certificate before establishing a connection, making spoofing significantly harder. The residual risk considerations mirror those of Threat 4.

An attacker may attempt to append their own certificate to a trust list in transit (Threat 24), achieving **TC1**. This is the same as Threat 7, just on a different link, so same mitigations and risk considerations apply.

Incident analysis on this link is hindered if the GDS keeps no record of interactions (Threat 25), leading to **TC5**. This is mitigated by M11, with residual risk considerations matching those of Threat 9.

An attacker may intercept certificates or trust lists in transit (Threat 26) to learn about the system configuration, thereby achieving **TC4**. Mitigation is provided by M9, with risk considerations similar to Threat 10.

As with other links, a DoS attack is possible: an attacker may drop packets or inject oversized payloads to disrupt communication and incapacitate either the Registrar or the GDS (Threat 27), targeting **TC2** and **TC3**. Mitigations M12 and M13 apply here as well, with residual risk considerations parallel to those in Threat 11.

Finally, an elevation of privilege may occur if an internal entity other than the Registrar attempts to communicate with the GDS (Threat 28), potentially

enabling device onboarding while bypassing attestation (**TC1**). This is mitigated by M14, with residual-risk considerations matching those of Threat 13.

To summarize, some threats can be mitigated by relying on the OPC UA security stack, such as DoS, certain tampering threats, and information-disclosure threats. However, it is not enough to cover everything: additional solutions are necessary for best protection. For best mitigation of the severe **TC1**, HSMs in combination with secure boot and runtime measurement are required. Careful configuration of *trust on first use* behavior on the attester and an access control system would further minimize the residual risks.

6 Conclusion and Future Work

In this paper, we proposed a method for integrating remote attestation into the OPC UA onboarding process. We began by defining the prerequisites a device must meet during onboarding, namely possessing the correct *identity* and being in a healthy *state*. If both aspects can be verified, the device can be considered *trustworthy*. We then examined how OPC UA currently handles onboarding and observed that it verifies identity, but not state. To address this gap, we extended the model to include a state-verification step via remote attestation. Our solution is based on the RATS reference interaction models [7], and we provided a reference implementation demonstrating how such an integration can be realized in practice. In addition, we conducted a threat analysis of the proposed attestation-enhanced onboarding workflow to identify potential vulnerabilities in the state-verification step and outlined several mitigations that improve the robustness of the overall process. We show that while the OPC UA security stack provides baseline protection, it is insufficient on its own to mitigate all identified threats.

Future work aims at exploring more complex attestation scenarios in which some devices are located behind isolated or segmented networks and cannot be reached directly. Moreover, the reference implementation is extended with a more realistic attesting environment, such as physical hardware with a TPM or an emulated TPM, as well as a more comprehensive system for managing reference values and endorsements within the onboarding workflow.

Acknowledgments. This work is supported by the Swedish Knowledge Foundation (KKS) via the SEINE and MARC projects and by the Swedish Governmental Agency for Innovation Systems (VINNOVA) via the FLEXATION and iSecure projects.

References

1. ABB: Modular automation - how it's being used and how you can take advantage of it. ABB (2024), https://new.abb.com/control-systems/modular-automation
2. Ambrosin, M., et al.: Collective remote attestation at the internet of things scale: State-of-the-art and future challenges. IEEE Communications Surveys & Tutorials **22**(4), 2447–2461 (2020)

3. Asim, M., et al.: SecT: A zero-trust framework for secure remote access in next-generation industrial networks. IEEE Journal on Selected Areas in Communications **43**(6), 2293–2311 (2025)
4. Bhole, M., et al.: From manual to semi-automated safety and security requirements engineering: Ensuring compliance in Industry 4.0. In: 50th Annual Conference of the IEEE Industrial Electronics Society (2024)
5. Birkholz, H., et al.: Remote ATtestation procedureS (RATS) Architecture. RFC 9334 (Jan 2023), https://www.ietf.org/rfc/rfc9334.html
6. Birkholz, H., et al.: RATS Conceptual Messages Wrapper (CMW). Internet-Draft draft-ietf-rats-msg-wrap-21, IETF (Nov 2025), https://datatracker.ietf.org/doc/draft-ietf-rats-msg-wrap/21/
7. Birkholz, H., et al.: Reference Interaction Models for Remote Attestation Procedures. Internet-Draft draft-ietf-rats-reference-interaction-models-15, IETF (Nov 2025), https://datatracker.ietf.org/doc/draft-ietf-rats-reference-interaction-models/15/
8. Birnstill, P., et al.: Introducing remote attestation and hardware-based cryptography to OPC UA. In: 22nd IEEE International Conference on Emerging Technologies and Factory Automation (2017)
9. Howard, M., Lipner, S.: The Security Development Lifecycle. Microsoft Press (2006)
10. IEC: IEC 62541: OPC UA (2025), International Standard, multi-part
11. IEEE: IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity. IEEE Std 802.1AR-2018 (Revision of IEEE Std 802.1AR-2009) (2018)
12. Juhola, A., Kylänpää, M.: Experimental implementation of remote attestation over OPC UA protocol. In: 2022 International Conference on Networks, Communications and Information Technology (CNCIT). pp. 83–88 (2022)
13. Khan, R., et al.: Stride-based threat modeling for cyber-physical systems. In: IEEE PES Innovative Smart Grid Technologies Conference Europe (2017)
14. Kohnhäuser, F., et al.: Secure onboarding of IIoT devices using OPC UA. In: 27th IEEE International Conference on Emerging Technologies and Factory Automation (2022)
15. Leander, B., et al.: Redundancy link security analysis: An automation industry perspective. In: IEEE 29th International Conference on Emerging Technologies and Factory Automation (2024)
16. Matischek, R., Bara, B.: Application study of hardware-based security for future industrial IoT. In: 22nd Euromicro Conference on Digital System Design (2019)
17. Meier, D., et al.: Secure provisioning of OPC UA applications using the asset administration shell. In: 17th Conference on Industrial Electronics and Applications (2022)
18. Red Hat: Attestation in confidential computing. Red Hat Blog (May 2023), accessed: 2025-10-12
19. Shelby, Z., et al.: The constrained application protocol (coap). Request for Comments 7252, IETF (Jun 2014), https://www.rfc-editor.org/rfc/rfc7252
20. Usman, A.B., Asplund, M.: Remote attestation with software updates in embedded systems. In: IEEE Conference on Communications and Network Security (2024)
21. ZVEI—German Electrical and Electronic Manufacturers' Association: Process Industrie 4.0: The age of modular production. Tech. rep., ZVEI (2019)