Zero-Touch Mutual Authentication Scheme for 6TiSCH Industrial IoT Networks

Ali Haj-Hassan*†, Youcef Imine†, Antoine Gallais†, Bruno Quoitin*
* Computer Science Department, University of Mons, Belgium
† Univ. Polytechnique Hauts-de-France, LAMIH, CNRS, UMR 8201, INSA Hauts-de-France, France
ali.hajhassan@uphf.fr youcef.imine@uphf.fr antoine.gallais@uphf.fr bruno.quoitin@umons.ac.be

***Abstract*—Industrial IoT (IIoT) networks must provide reliability, determinism and security. In terms of security, ensuring an efficient authentication is still a challenging task. Most of the authentication approaches adopted in IIoT wireless communication protocols rely on the existence of a pre-shared key (PSK) between each joining node and the central authority of the network. How to share the PSK is however not specified in their standards.**

In this paper, we propose a new zero-touch mutual authentication and key establishment protocol for IIoT. In our protocol, the network coordinator authenticates a new joining node using certificates. Then, the joining node authenticates the network coordinator through a novel consensus achieved among nodes that are already in the network. Our novel consensus is based on Shamir secret sharing, and allows each new node to build its trust based on the knowledge of a group of nodes that are already in the network. Finally, our protocol allows to securely establish a common key between new joining nodes and the network coordinator over a public channel. In addition to the theoretical aspects, we evaluate the performance of our protocol under two attack scenarios where 33% of the nodes in the network are malicious. We show in all cases that we efficiently ensure authentication with high success probabilities, by establishing a consensus including a limited number of nodes in the network.

***Index Terms*—Industrial Networks, 6TiSCH, IoT Security, Key Establishment, Authentication, Consensus**

I. Introduction

The convergence of low power wireless networks and the IP-enabled networks changed the perspectives of IoT. Today, IoT devices can communicate between themselves and can connect to other IP networks. The integration of this technology with Industry 4.0 led to the existence of a new field called Industrial IoT (IIoT). As an example, IIoT would allow the cooperation between production chains distributed among different sites. To make this possible, IIoT must satisfy two main requirements: reliability and security. Indeed, continuous production relies on uninterrupted communications. However, enabling IP connectivity increases the attack surface. Malicious activities could cause damages to the infrastructure, disrupt the production and even put the life of workers on risk.

WirelessHART and ISA100.11a are two protocols well adopted in industries, allowing the convergence of low-power wireless devices and the Internet Protocol. Recently, the IETF introduced a promising IIoT solution called 6TiSCH [1] to enable high reliability IPv6 wireless sensor networks. It is based on the IEEE 802.15.4 Time Slotted Channel Hopping (TSCH) medium access control which combines time division multiplexing, to make access deterministic, and frequency hopping, for increased robustness against interferences.

A joining protocol defines how new nodes join a 6TiSCH network. In this phase, a new joining node exchanges join messages with a network coordinator, responsible of managing the network, and giving access to new nodes. The join requests are forwarded to the coordinator through nodes that are already in the network, playing an intermediate role in this phase.

In the absence of security measures during the joining phase, malicious nodes can join the network and disturb forthcoming joining events. Moreover, since a new joining node has no previous knowledge on the network, it has no way to verify the correctness of information during this join phase. Therefore, it may be subject of impersonation attack led by malicious nodes. Indeed, malicious nodes may forward the node's join requests to another malicious node pretending to be the network coordinator, instead of forwarding it to the true one.

In order to overcome these issues, it is essential to secure the joining phase with a mutual authentication allowing: (1) both the network coordinator and the new joining node to verify each other and (2) to establish a common key to secure forthcoming exchanges between the node and the coordinator.

Many existing works studied the authentications and key establishment concerns in IoT networks [2, 3]. Most of these protocols are either based on pre-shared keys, shared physical measurement (e.g., movement, temperature) or certification-based authentication. In the case of pre-shared keys based protocol, manual configuration of each expected joining node is required. However, this approach is not efficient in dynamic large scale networks. On the other hand, protocols based on sharing a physical measurement may not be accurate enough due to the instability of physical environments in an industrial context. Finally, certification based solutions are not well suited for mutual authentication in the IoT since devices would need to store multiple chains of trust, which is not compatible with their constrained storage capabilities.

In this paper, we propose a novel zero-touch mutual authentication protocol for IIoT networks. On one hand, our solution is based on certificates to allow the network coordinator to authenticate new joining nodes. On the other hand, we propose a new consensus approach among network members, based on Shamir secret sharing, allowing new joining nodes to authenticate and establish keys with the coordinator. Our dynamic and scalable joining scheme has the following features:

- Mutual authentication without a pre-configured key;
- Low storage overhead since the nodes only have to store few pieces of information to run the whole protocol;

- Easy integration with communication protocols adopted in IoT networks, demonstrated by our application to the joining phase of 6TiSCH.

The remaining of this paper is structured as follows. In section II, we discuss the related work of authentication schemes in IoT. In section III we a give background on 6TiSCH protocol and Shamir secret sharing. We present our solution in section IV. Then, we describe its attack model and security analysis in sections V and VI respectively. In section VII we provide a performance evaluation. Finally, we conclude in section VIII.

II. Related Work

In most IoT wireless networks, authentication schemes are based on pre configuring a new joining node with a pre-shared key (PSK), before the first phase of authentication. For instance, each device has a unique identifier and needs to pre-share a symmetric key with the network coordinator, which can further authenticate known devices upon new communication [4]. This requires a prior configuration of all nodes that will join the network. In 6TiSCH protocol, the authors of the minimal security draft of the IETF [5] assumes that the exchanges during the joining phase of a new node is secured using a PSK. However, they do not describe how this key was shared. The same assumption is made in [6], as the authors proposed a 3-way mutual authentication mechanism based on a multi key called secure vaults. Here, the PSK consists of the secure vault shared with the IoT devices during the deployment phase. This secure vault will be used by the coordinator to challenge a device and authenticate it. Other PSK-based authentication schemes were proposed in order to authenticate a joining node, for wireless network protocols like 6LoWPAN and LoRaWAN [7–9]. All these solutions require pre-configuring the IoT devices before initiating this phase, which is not efficient for the case of a large scale dynamic industrial network.

Certificate based authentication have been proposed for IoT networks as well. In [10], the authors proposed a two phase mutual authentication solution: (1) the registration phase where each edge device acquires its security credentials from a Certificate Authority (CA) and stores its chain of trust. (2) An authentication phase is then executed where devices establish a secure communication channel. However, this solution does not address scenarios with multiple CAs, which would require constrained devices to store a considerable number of chains of trust. Moreover, even though some solutions allow to assess the status of a given certificate (e.g. OCSP), the consistency of revocation lists may be endangered once large scale and lossy networks are considered, such as the heterogeneous IoT networks targeted here.

In [11], the authors proposed SHAKE, an authentication protocol that relies on data acquired from physical sensors. It consists in holding two nodes together and shaking them simultaneously. An accelerometer in both nodes captures this same movement and transforms the measured value into a shared secret key. Other physical environmental measurements have been proposed in the context of IoT to ensure mutual authentication [12, 13]. Meanwhile, such physical manipulations can be costly and hard to ensure in large-scale industrial environments. In addition, due the heterogeneity in industries, some nodes may be either too constrained to hold such sensor or impossible to move once installed.

III. Background

A. 6TiSCH Joining Phase

In a 6TiSCH network, new nodes need to go through an initial joining phase to get admitted by the network coordinator and obtain link-level security credentials.

The Constrained Join Protocol (CoJP) [5] handles the parameters distribution needed for new nodes to join a 6TiSCH network. In CoJP, three entities play a main role during the joining phase. (1) The Join Registrar/Coordinator (JRC) is the entity responsible for managing the network and giving access to new nodes. (2) The pledge is the node seeking to join the network. (3) The Join Proxy (JP) is a node already in the network that plays an intermediary in the exchanges between a pledge and the JRC.

During this phase different types of messages are exchanged:

- Enhanced Beacon (EB): message sent on a regular basis by JPs allowing pledges to discover the network and inviting them to join;
- Join Request: message sent from the pledge to the JRC, through the JP. It includes the role it requests to play in the network, as well as the identifier of the network it requests to join.
- Join Response: message sent from the JRC to the pledge through the JP. It contains different parameters needed by the pledge to become a fully operational network node.

B. Shamir Secret Sharing

Shamir's secret sharing method [14] consists in dividing a secret into parts, and share these parts, in a way that a minimum number of shares is needed to reconstruct the secret. For that, a finite field Z_p is adopted to generate elements used as coefficients for a polynomial $Q(x)$ of degree m. The secret here consists of the value $Q(0)$. To redefine this polynomial, a least $m+1$ points generated from this polynomial are needed. Given $m+1$ points, $\{P(x_i, y_i)\}, i \in \{1, .., m+1\}$, generated from a polynomial $Q(x)$, Lagrange method is used for polynomial interpolation as follows:

$$f(x) = \sum_{i=1}^{m+1} y_i \prod_{j=1, j\neq i}^{m+1} \frac{x - x_j}{x_i - x_j} \quad (1)$$

IV. Proposed Solution

In this section we detail our proposed solution for a mutual authentication between a new joining node and the network coordinator. Our solution assumes no pre-shared key previously configured by an operator. End devices must only contain a pre-installed certificate configured by the manufacturer.

In the following sections, we apply our solution to the joining phase of 6TiSCH (CoJP). We assume that the JRC is a fully trusted entity that manages the network security. To the opposite, JPs may act in a malicious way at this phase,

thus they cannot be fully trusted. We assume however that no more than one third of the nodes in the network are malicious, similarly to Byzantine fault tolerance assumption [15].

A. Main Idea

Our approach takes into consideration the imbalance of resources in the network. The JRC has more capabilities than the resource-constrained pledges. For this reason, we propose to use two different mechanisms to achieve mutual authentication.

At the first step, the JRC authenticates the pledge who asks to join the network. This phase relies on a certification-based authentication. Having the capability to access Certificate Authorities through chains of trust, the JRC can verify a certificate provided by a pledge.

In a second step, the pledge proceeds with the JRC authentication. It is based on a consensus: multiple JP existing in the network prove for the pledge the legitimacy of the JRC. This is done by revealing a secret shared between the JRC and the network's nodes using Shamir secret sharing (Sec. III-B). Hence, multiple JP contacted by the pledge, called edge JPs, must collaborate together in order to reveal this secret contained at the JRC. Therefore, a collusion of many JPs in the network points towards its secret.

Once both sides are mutually authenticated, a key is established between them for the subsequent exchanges.

B. Setup Phase

While bootstrapping the network, the JRC executes the following steps to prepare the parameters used by our protocol:

- Let G be a cyclic group of order p where p is a safe prime and g its generator.
- Let Sk_{JRC} and Pk_{JRC} a pair of private and public key respectively, used by the JRC and the nodes to sign and verify exchanged messages.
- Define two mapping functions $f1 : G \rightarrow Z_p$ and $f2 : Z_p \rightarrow G$ such that $\forall e \in G$, we have $f2(f1(e)) = e$.
- Let w be a random element in Z_p, compute the group key $S = g^w$.
- Let $Q(x) = a_0 + a_1x + a_2x^2 + ... + a_mx^m$ of degree m. The coefficients a_i, where $i \in [1, m]$, are random elements in Z_p and $a_0 = f1(S)$.
- The JRC creates a hashtable HT to register authentication session identifiers along with session keys for each new node willing to join the network.

In an industrial context, we certainly have nodes that are already in the network. Therefore, every node i existing in the network is configured by the coordinator with a tuple $(\sigma_i, P_i, Pk_{JRC})$ where

- $P_i = (x_i, y_i)$, where $y_i = Q(x_i)$, is a distinct random point generated from the secret polynomial $Q(x)$;
- σ_i is a signature of P_i as:

$$\sigma_i = Sign(H(x_i||',' ||y_i), Sk_{JRC})$$

where $Sign$ is a signature scheme and H is a hash function.

Along these minor modifications on 6TiSCH main protocol, we propose also to modify the Enhanced Beacon EB of 6TiSCH in order to contain the degree m of $Q(x)$ as well as a service Srv advertising the service of the network. Based on Srv, a node takes or not the EB into consideration.

C. Pledge Authentication

The JRC needs to authenticate the pledges looking to join the network. We propose to use a certificate-based authentication as the JRC has the necessary computational and storage resources. Since we consider an industrial context, we can assume that the JRC has prior knowledge of the type of nodes that may join its network. Therefore, the JRC only stores certificate chains of trust related to those types of nodes. Moreover, we can assume that in each pledge a certificate is installed by the manufacturer after production, as part of the non volatile data, and no other configuration is needed.

Our proposed certificate-based authentication of the pledge is established as follows:

- Edge JPs send EBs to the pledge on a regular basis.
- The pledge collects EBs advertising Srv. It does so for some time T, trying to get in touch with more JPs advertising Srv.
- The pledge sends its certificate, as a join request, to all contacted edge JPs.
- The edge JPs forward the request to the JRC.
- The JRC verifies the certificate and authenticates or not the pledge.
- In case it is authentic, the JRC saves in HT, a hash of the public key of the pledge $H(PK_{Pledge})$, as an authentication session identifier.
- The JRC allows edge JPs to continue the communication with this pledge by sending back a response containing the pledge's public key PK_{Pledge}.

Note that, proceeding in the execution of the protocol using PK_{Pledge} to encrypt the further exchanges: (1) proves that the pledge owns its corresponding secret key, and (2) allows at the same time to secure the communication with the Edge JPs.

D. JRC Authentication

In this section we present the whole JRC authentication. Given a set $S_{E_{JP}} = \{E_{JP_1}, E_{JP_2}, ..., E_{JP_N}\}$ of N edge JPs to contact, the JRC goes through the following steps:

- The pledge requests from each edge JP E_{JP_i} a packet $P_{JP_i} = \{P_{i,1}, P_{i,2}, ..., P_{i,m}\}$ of m points where $P_{i,k} = (x_k, y_k), k \in [1, m]$.
- Each E_{JP_i} receiving the pledge request, asks for $m - 1$ distinct points $P_{i,k}$ from random nodes in the network.
- Each node j receiving E_{JP_i}'s request, sends its own point $P_j = (x_j, y_j)$ and the signature σ_j provided by the JRC.
- For each received (P_j, σ_j), the edge JP verifies the signature σ_j using the JRC's public key Pk_{JRC}.
- The edge JP E_{JP_i} forms a packet of points P_{JP_i} which contains the collected points (verified in the previous step), as well as E_{JP_i}'s own point.
- The edge JP sends P_{JP_i} to the pledge encrypted with PK_{Pledge}.

- The pledge decrypts the packets with its private key SK_{Pledge} and form a set $P = \{P_{JP_1}, P_{JP_2}, ..., P_{JP_N}\}$ coming from N received edge JPs.
- The pledge constructs a set $C_X = \{C_1, C_2, ..., C_X\}$ of all possible combinations in P, where $X = C^2_{n(P)}$ is the number of combinations without repetition of two sets of points $P_{JP_i} \in P$.
- $\forall\ C = \{P_{JP_i}, P_{JP_j}\}$ where $P_{JP_i}, P_{JP_j} \in P$, define:
$$SC_i = P_{JP_i} \cup P_{JP_j} = \{P_{i,1}, ..., P_{i,m}, P_{j,1}, ..., P_{j,m}\}$$
- Let $FSP_i \subset SC_i$ be a subset of $m+1$ distinct points randomly chosen from each SC_i, defined as:
$$FSP_i = \{P_1 = (x_1, y_1), ..., P_{m+1} = (x_{m+1}, y_{m+1})\}$$
- FSP_i is used to reconstruct the secret $Q_i(0)$ using Lagrange polynomial interpolation as follows:
$$Q_i(0) = \sum_{k=1}^{m+1} y_k \prod_{z=1, z\neq k}^{m+1} \frac{-x_z}{x_k - x_z} \quad (2)$$

We note that the polynomial interpolation (2) is based on the points collected from nodes already active on the network. We recall that these points were initially provided by the JRC and generated from the polynomial $Q(x)$ defined in the setup phase. Therefore, an interpolation polynomial on these points should provide the same value $Q(0)$ defined by the JRC.

Considering all the combinations done on the points collected by the pledge and provided by the JRC, we will end up with $X = C^2_{n(P)}$ repetitive values $Q_i(0) = Q(0)$ computed during the previous step.

Being able to repeatedly retrieve the same value $Q(0)$ allows to achieve a consensus through multiple JPs in the network, directing the pledge towards the JRC's identity.

Nevertheless, malicious edge JPs may be among the edge JPs collecting points. Obviously malicious edge JPs aim to deviate the interpolation's results, leading to some values $Q_i(0) \neq Q(0)$ among the X interpolations performed by the pledge. Whereas, as long as the rate of malicious nodes in the network is up to 33%, the consensus can be achieved by considering the value the most frequent among the X interpolations. Hence, the success of the consensus is based on the majority of honest nodes contacted during the joining phase.

E. Key Establishment

The JRC authentication phase allowed the pledge to discover $Q(0)$. Therefore, the pledge proceeds to verify and establish a common key with the JRC through the following procedure based on the Diffie-Hellman exchange:

- The pledge retrieves the group key as:
$$S' = f2(Q(0)) = f2(f1(S)) = S$$
- The pledge generates the following parameters: a random element r in Z_p, a challenge consisting of a random series of bits $C \in \{0,1\}^*$ and a random element $El \in G$;
- The pledge sets the session key $CS = H(El)$ and computes $CT = enc(C, CS)$, where $enc()$ is a symmetric encryption algorithm;
- The pledge calculates $Sig = Sign(H(g^r || S^r \cdot El || CT), SK_{Pledge})$ where $Sign$ is a secure signature scheme, H is a hash function and SK_{Pledge} is the private key of the pledge;
- The pledge sends $(g^r, S^r \cdot El, CT, Sig, PK_{Pledge})$ to JRC;
- The JRC receiving PK_{Pledge}, retrieves the authentication session by looking up for $H(PK_{Pledge})$ in HT. Then, it verifies the signature Sig using PK_{Pledge};
- We recall that the group key has been generated as $S = g^w$ where $w \in Z_p^*$ was a random value chosen by the JRC during the setup phase. Thus, the JRC calculates:
$$El' = (S^r \cdot El)/(g^r)^w = El$$
- The JRC recovers $C = dec(CT, H(El'))$, where $dec()$ is a symmetric decryption algorithm;
- The JRC sends recovered C to the pledge as a response to the challenge and saves $H(El')$ as a session key for the pledge's session identifier in HT;
- The pledge considers the key establishment has succeeded if it receives the response C before a time T.

V. Attack Models

In this section, we predict the attacks that may happen during the authentication phase. The main vulnerability that we may have is the existence of malicious nodes in the network, controlled by remote nodes, or programmed to perform malicious behaviours. Once acting as edge JPs, these malicious nodes may have two purposes, either fail the authentication or lead the pledge to authenticate a wrong JRC. For that, a malicious node can conduct these types of attacks:

- **Individual attack:** A pledge requests an edge JP to collect points from other JPs in the network in order to reconstruct the polynomial and reveal the secret. A malicious edge JP creates its own polynomial and generates the requested number of points from it, instead of contacting other JPs, trying to fail the interpolation done by the pledge.
- **Collaborative attack:** All malicious edge JPs in the network have a sort of agreement between themselves. They have the same polynomial from which they generate points to be sent to the pledge. Hence, they all work together looking for leading the interpolation done by the pledge to one wrong JRC.
- **Impersonation attack:** An edge JP tries to impersonate the pledge or the JRC while playing the role of intermediate between them.

Note that other types of attacks threatening a multi-hop wireless network, such as jamming attack or nodes selfishness, etc., are not considered in the scope of this paper.

VI. Security Analysis

In this section, we analyse the capability of our solution to face the possible attacks. We recall that in order to authenticate, the pledge requests packets of points from multiple edge JPs, and combines these packets two by two to achieve a consensus. The existence of malicious nodes between contacted edge JPs leads to fail the calculation for some of these combinations.

In the worst case scenario, where the third of these edge JPs are malicious nodes, we have for any number of edge

JPs $N>3$, three categories of combinations of packets: (1) a combination of packets collected by two honest edge JPs; (2) a combination of a packet collected by an honest edge JP and another packet collected by a malicious one, and (3) a combination of packets collected by two malicious edge JPs.

For the first category, the calculation always leads to the correct secret. For the second one, each calculation leads to a wrong secret that is not repeated in the other calculations. For the third category, the calculation results depends on the type of attack conducted. In case of an individual attack, a different wrong secret is calculated for each different combination. Whereas, in case of a collaborative attack, the same wrong secret is calculated for all the combinations of nodes belonging to this category.

In terms of percentage of repetition of each secret resulting from the combinations, we notice that in the first category of combinations we have $C^2_{\frac{2\cdot N}{3}}/C^2_N$. Moreover, the percentage of each secret found in the second category is $1/C^2_N$.

On the other hand, as secrets founds in the third category may be repeated or not depending on the attack scenario, we have, in an individual attack, the percentage of each secret resulting from the combination is $1/C^2_N$. Whereas, this percentage is equal to $C^2_{\frac{N}{3}}/C^2_N$ in the case of a collaborative attack.

In all cases, we clearly see that the percentage of the repetition of the secret found through combinations of the first category always represent the majority. Therefore, even under individual and collaborative attacks scenarios, our protocol always achieves the consensus for any number of edge JPs $N > 3$ as long as the rate of malicious edge JPs does not exceed $N/3$.

For the impersonation attack during the key establishment phase, a malicious edge JP cannot impersonate the pledge since the parameters sent by the pledge during this phase are signed with its private key. Likewise, a malicious edge JP cannot impersonate the JRC since it needs to be able to recover the value w (known only by the JRC), given $S = g^w$. This is necessary to get the session key and to send back the challenge waited by the pledge. However, succeeding to lead this attack is equivalent to solving discrete logarithm problem known to be hard in multiplicative cyclic groups.

VII. Performance Evaluation

In order to evaluate the performance of our protocol, we developed an ad-hoc simulator in Java. In our simulator, we consider one JRC, multiples JPs and plegdes. Each one of these entities is implemented as a thread executing its tasks defined in our protocol and communicating with other entities through TCP sockets. We also use a lightweight configuration adapted for constrained objects in JPBC security library to implement cryptographic operations. JPs can be either honest or malicious. If a JP is malicious, it can be configured to perform an individual or a collaborative attack (Section V). The edge JPs executing the consensus are randomly chosen among the JPs launched in our simulation. The simulation parameters (Table I) are varied in order to evaluate their impact on the success rate of the authentication. The results are presented in the following sections.

Total number of nodes	100
Degree m of Q(x)	[2-10], default value=2
Rate of malicious nodes	{10,20,33}%, default value=33%
Simulation rounds	1000

TABLE I: Simulation parameters.

A. Security Robustness

Figure 1 represents the variation of authentication success rate according to the number of edge JPs, for different rates of malicious nodes conducting an individual attack. As we can see, in the worst case (33% of malicious nodes in the network), we reach a rate of 0.8 of successful authentications starting from a number of edge JPs equal to 5. This value starts to converge to 1 starting from a number of edge JPs equal to 8 and lesser if we consider a rate of 10% or 20% of malicious nodes. Note that for a rate of malicious nodes in the network higher than 33%, to attain the same success rate, higher number of edge JPs must be contacted. Since this solution is proposed for an industrial context, we consider that the availability of edge JPs is not a constraint. Moreover, for an individual attack, the cases where the authentication does not succeed represent the cases where the consensus has not been achieved, not that an attack has succeeded.

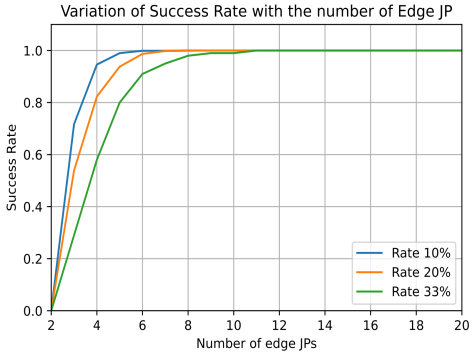

Fig. 1: Variation of authentication success rates in individual attacks.

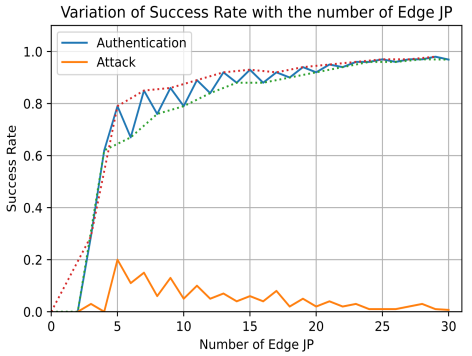

Fig. 2: Variation of authentication success rate in collaborative attacks.

Figure 2 represents the same variation as Figure 1 in a scenario where malicious nodes (33% of the network) conduct a collaborative attack. Although we notice the convergence of authentication success rates, we also observe an instability in the rates achieved in the case where we have an even and an odd number of edge JPs. Indeed, if we separately consider the success rates curve of even numbers of edge JPs (green curve) and odd numbers (red curve), we notice that the rate in both curves is increasing. However, the convergence of the red curve

is much quicker than the green one. To explain this behavior, we calculated (in Table II) the rate of how many times, in average, the number of malicious edge JPs has exceeded the half of the contacted edge JPs during the simulation. We note that in the collaborative attack, when the number of malicious edge JPs exceeds the half of contacted edge JPs, the authentication fails since we achieve a consensus leading to malicious nodes' secret. Now if we check the rates in table II, we can clearly notice that we exceed more often the half when an even number of edge JPs is contacted compared to when an odd number of edge JPs is contacted. This means that, the collaborative attack succeeds more for even numbers than odd numbers. Therefore, the convergence of authentication success rate becomes slower for even numbers as noticed in Figure 2.

# edge JPs	2	3	4	5	6	7	8	9	10
Avg. Exceed Time	0.55	0.26	0.37	0.2	0.32	0.15	0.23	0.13	0.2

TABLE II: Percentage of times malicious exceeding half edge JPs.

B. Communication and Energy Efficiency

In this section, we evaluate the trade-off between the robustness of our protocol and the energy efficiency. Given a polynomial of degree m and a number N of edge JPs to be contacted to collect $m-1$ points each, the number of messages N_{msg} exchanged during one JRC authentication is:

$$N_{msg} = (m-1) \times N \times 2 + N \times 2 = 2 \times m \times N \qquad (3)$$

As we can see, the energy consumed in terms of communication depends on the degree m and the number of edge JPs to contact.

In Table III, we represent the impact of the degree m on authentication success rates and the number of messages exchanged. As we can notice, increasing the value of m increases the number of messages exchanged but does not have an impact on success rates. Thus, bigger values of m do not make our protocol more robust. Thus, we can reduce the communication overhead by adopting the smallest value of m.

Degree m	2	3	4	5	6	7	8	9	10
Success Rate	0.80	0.79	0.80	0.79	0.77	0.77	0.77	0.78	0.77
# Messages	20	30	40	50	60	70	80	90	100

TABLE III: Degree m modification with five edge JPs.

VIII. Conclusion

In this paper, we propose a zero-touch mutual authentication for 6TiSCH industrial IoT networks. Our scheme is multi fold: first the network coordinator authenticates new joining nodes using certificates. Second, a novel consensus among the network's nodes, and based on Shamir shared secret, attests the coordinator's identity. Finally, a session key is established between these two entities. Unlike most of existing solutions, our scheme does not require any prior configuration of new nodes. Theoretical analysis and simulations are conducted to evaluate the robustness of our protocol in attack scenarios assuming the existence of up to 33% of malicious nodes in the network. The results of the simulation are consistent with the theoretical analysis, and have shown the efficiency of our protocol and its resilience against attacks. In the future, we intend to investigate the performance of our protocol in the case where the joining phase is performed in a network zone having a small nodes concentration. We also intend to simulate a real IIoT scenario, where we consider different network topologies, to get a better evaluation in terms of energy consumption.

References

[1] X. Vilajosana et al., "IETF 6TiSCH: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 22:1, 2019.

[2] M. El-Hajj et al., "A survey of Internet of Things (IoT) authentication schemes," *Sensors*, vol. 19, no. 5, 2019.

[3] Y. Yang et al., "A survey on security and privacy issues in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, 2017.

[4] M. A. Jan et al., "A robust authentication scheme for observing resources in the Internet of Things environment," in *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, 2014, pp. 205–211.

[5] M. Vučinić, J. Simon, K. Pister, and M. Richardson, "Constrained Join Protocol (CoJP) for 6TiSCH," RFC 9031, May 2021.

[6] T. Shah et al., "Authentication of IoT device and IoT server using secure vaults," in *IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom)*, 2018.

[7] R. Sanchez-Iborra et al., "Enhancing LoRaWAN security through a lightweight and authenticated key management approach," *Sensors*, vol. 18, no. 6, p. 1833, 2018.

[8] H. R. Hussen et al., "SAKES: Secure authentication and key establishment scheme for M2M communication in the IP-based wireless sensor network (6LoWPAN)," in *International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2013, pp. 246–251.

[9] A. Esfahani et al., "A lightweight authentication mechanism for M2M communications in industrial IoT environment," *IEEE Internet of Things Journal*, no. 1, 2017.

[10] P. Porambage et al., "Two-phase authentication protocol for wireless sensor networks in distributed IoT applications," in *IEEE Wireless Comm. and Netw. Conf.*, 2014.

[11] E. Bejder et al., "SHAKE: SHared Acceleration Key Establishment for Resource-Constrained IoT Devices," in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*. IEEE, 2020, pp. 1–6.

[12] Z. Li et al., "Secret key establishment via RSS trajectory matching between wearable devices," *IEEE Transactions on Information Forensics and security*, vol. 13, no. 3, pp. 802–817, 2017.

[13] A. Arno et al., "Accelerometer assisted authentication scheme for smart bicycle lock," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE, 2015.

[14] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, p. 612–613, Nov. 1979.

[15] L. Lamport et al., "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.