Chinese Journal of Electronics
vol. 33, no. 4, pp. 1046–1062, July 2024
https://doi.org/10.23919/cje.2023.00.192

RESEARCH ARTICLE

A General Authentication and Key Agreement Framework for Industrial Control System

Shan GAO[1], Junjie CHEN[1,2], Bingsheng ZHANG[1], Kui REN[1,3], Xiaohua YE[4], and Yongsheng SHEN[4]

1. *School of Cyber Science and Technology, Zhejiang University, Hangzhou 310058, China*
2. *Hangzhou Global Scientific and Technological Innovation Center, Zhejiang University, Hangzhou 311200, China*
3. *Zhejiang Provincial Key Laboratory of Blockchain and Cyberspace Governance, Hangzhou 310058, China*
4. *Hangzhou City Brain Co., Ltd, Hangzhou 310024, China*

Corresponding author: Junjie CHEN, Email: 22121095@zju.edu.cn
Manuscript Received May 23, 2023; Accepted November 10, 2023

Abstract — In modern industrial control systems (ICSs), when user retrieving the data stored in field device like smart sensor, there exists two main problems: one is lack of the verification for identification of user and field device; the other is that user and field device need exchange a key to encrypt sensitive data transmitted over the network. We propose a comprehensive authentication and key agreement framework that enables all connected devices in an ICS to mutually authenticate each other and establish a peer-to-peer session key. The framework combines two types of protocols for authentication and session key agreement: The first one is an asymmetric cryptographic key agreement protocol based on transport layer security handshake protocol used for Internet access, while the second one is a newly designed lightweight symmetric cryptographic key agreement protocol specifically for field devices. This proposed lightweight protocol imposes very light computational load and merely employs simple operations like one-way hash function and exclusive-or (XOR) operation. In comparison to other lightweight protocols, our protocol requires the field device to perform fewer computational operations during the authentication phase. The simulation results obtained using OpenSSL demonstrates that each authentication and key agreement process in the lightweight protocol requires only 0.005 ms. Our lightweight key agreement protocol satisfies several essential security features, including session key secrecy, identity anonymity, untraceability, integrity, forward secrecy, and mutual authentication. It is capable of resisting impersonation, man-in-the-middle, and replay attacks. We have employed the Gong-Needham-Yahalom (GNY) logic and automated validation of Internet security protocols and application tool to verify the security of our symmetric cryptographic key agreement protocol.

Keywords — Industrial control system, Authentication and key agreement framework, Lightweight key agreement protocol, Automated validation of Internet security protocols and application.

Citation — Shan GAO, Junjie CHEN, Bingsheng ZHANG, *et al.*, "A General Authentication and Key Agreement Framework for Industrial Control System," *Chinese Journal of Electronics*, vol. 33, no. 4, pp. 1046–1062, 2024. doi: 10.23919/cje.2023.00.192.

I. Introduction

In recent years, industrial control system (ICS) has garnered more and more attention from both industrial and academia communities. ICS is a complex distributed system extensively utilized in diverse industrial scenarios such as power plants, gas pipelines, and water plants. It plays an vital role in factory operations that enables operators to control and manage industrial processes effectively. Traditional ICS primarily prioritized system stability, often overlooking the need for robust security measures. However, with the advent of the industrial Internet of things (IIoT), modern ICS can now be accessed through open networks. While this facilitating remote management of industrial processes, it also exposes ICS to potential cyber-attacks, significantly increasing the risk of system failure. In practice, several prominent ICSs have suffered server damage due to cyber-attack. For instance, in 2012, the Stuxnet virus [1] infiltrated and reprogrammed industrial systems controlling centrifuges

Associate Editor: Prof. Jianfeng MA, Xidian University.

at the Iran nuclear power plant. Similarly, in 2021, hackers breached the Colonial Pipeline, resulting in a shutdown lasting six days in response to the cyber-attack [2]. Therefore, enhancing security measures has become imperative for modern ICS [3]. Ensuring entity authentication and preserving the confidentiality and integrity of the transmitted messages are critical requirements in this regard.

As depicted in Figure 1, a typical ICS architecture consists of four layers. The corporate network layer is connected to open networks, while the supervisory control layer encompasses an authentication server AS, engineer stations, and operator stations. This layer employs a firewall to filter incoming traffic. The AS assumes a vital role in entity authentication and key agreement within the ICS, enabling secure communication between ICS devices. Furthermore, the basic control layer comprises programmable logic controllers (PLCs) that establish a direct interface with field devices at the physical process layer using field buses. At the lowest layer, multiple industrial field devices operate. Field devices often possess limited resources in terms of memory space, computation capability, and communication bandwidth. Consequently, lightweight computations are more suitable for field devices than resource-intensive cryptographic operations. In general, the devices within corporate network require access to the data stored in field devices. To ensure the authenticity of message sources and preserve data confidentiality, it becomes imperative to employ a mutual authentication mechanism and key establishment scheme.

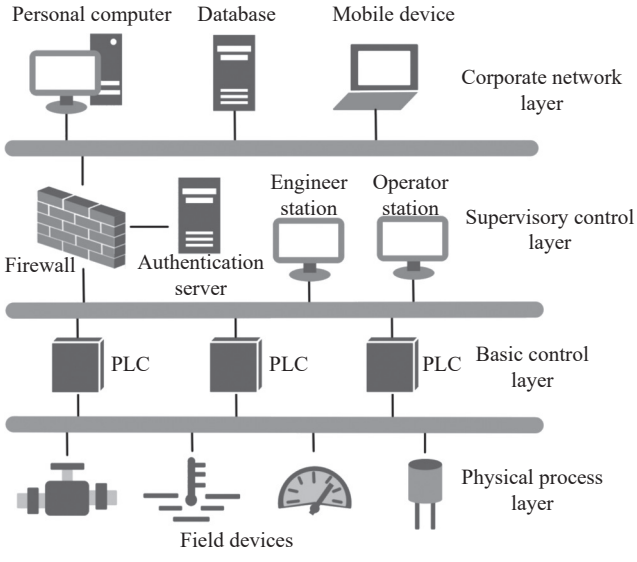

Figure 1 ICS architecture.

In this paper, we proposed a comprehensive authentication and key agreement framework for ICS that enables all connected devices to mutual authenticate each other and establish secure communication. Specifically, for the devices over the open Internet, we employ an asymmetric cryptographic key agreement scheme, leveraging the transport layer security (TLS) handshake protocol. For field devices within the internal network, we introduce a newly designed lightweight key agreement protocol that achieves mutual authentication and key agreement between the PLC and field devices.

Our contributions In this study, we have designed an authentication and key agreement framework specially tailored for ICS networks. Our contributions can be summarized as follows:

- We propose a hybrid authentication and key agreement framework that utilizes an asymmetric cryptographic key agreement scheme for devices connected to the open Internet, while employing a lightweight symmetric cryptographic key agreement scheme for field devices with limited resources within the internal network.
- We have designed a lightweight authentication and key agreement protocol that relies solely hash functions and exclusive-or (XOR) operations. Through comprehensive analysis, our protocol demonstrates resilience against impersonation attack, man-in-the-middle (MitM) attack, and replay attack. It also provides essential security features, such as identity anonymity, untraceability, message integrity, forward secrecy, and mutual authentication.
- The security of our lightweight protocol has been rigorously examined by using Gong-Needham-Yahalom (GNY) logic and automated validation of Internet security protocols and application (AVISPA). Additionally, our protocol demonstrates superior performance compared to or is comparable to the protocols proposed in [4]–[9] in terms of computation cost and communication overhead. The field device only needs to execute 4 hash functions during the authentication process. The simulation results indicate that the authentication process can be completed within 0.005 ms.

Roadmap The structure of this paper is as follows. Section II provides an overview of the related work in the field. In Section III, we present two cryptography algorithms, the system model, and security requirements. The details of our proposed framework are discussed in Section IV, while Section V presents a comprehensive security analysis of the proposed lightweight protocol. The performance evaluation of our protocol is presented in Section VI. Finally, we conclude our work in Section VII.

II. Related Work

1. User authentication scheme

In wireless sensor networks (WSNs), user authentication is essential to verify the identity of users before a sensor node transmits data to them, whether on-demand or upon event detection. The gateway node (GWN) primarily handles this authentication process. However, traditional user authentication solutions face significant challenges in WSN due to resource limitations. To overcome these challenges, Wong *et al.* [10] proposed a

lightweight authentication scheme based on hash functions and XOR operations. Despite its lightweight nature, their scheme exhibited vulnerabilities such as replay and forgery attacks. In response, Tseng *et al.* [11] improved Wong *et al.*'s scheme by enhancing security and introducing a password-changing phase to regularly update user passwords. Subsequently, Das [12] identified additional vulnerabilities in Wong *et al.*'s scheme, specifically the same login-ID threat and stolen-verifier attack. Das addressed these weaknesses by introducing a new user authentication protocol. However, Das's scheme had its own shortcomings, including the inability to change passwords, lack of mutual authentication, susceptibility to GWN bypassing attacks, and privileged-insider attacks. Several subsequent works [13]–[16] have further improved upon Das's scheme. Nonetheless, these user authentication schemes remain vulnerable to various attacks and primarily focus on preventing unauthorized access rather than ensuring data confidentiality and integrity.

2. Authentication and key agreement scheme

To address the challenges in user authentication schemes, lightweight protocols have been proposed to enable secure session key negotiation and achieve mutual authentication among the user, GWN, and sensor node. Turkanović *et al.* [17] introduced a lightweight authentication and key agreement protocol that offers secure key establishment, mutual authentication, password protection, and resilience against various attacks such as replay attacks, MitM attacks, and impersonation attacks. However, their scheme is vulnerable to impersonation attacks with node capture, stolen smart card attacks, sensor node spoofing attacks, stolen verifier attacks, and lacks backward secrecy. To overcome these limitations, Chang and Le [18] designed an advanced lightweight scheme that addresses the vulnerabilities in Turkanović *et al.*'s protocol. They incorporated elliptic curve cryptography (ECC) to provide perfect forward secrecy. However, Chang and Le's protocol is susceptible to offline password guessing attacks, user untraceability attacks, smart card recovery attacks, known session-specific temporary information attacks, and previous session key attacks. Amin *et al.* [19] identified vulnerabilities in Chang and Le's scheme and proposed a more realistic architecture for low-power sensor nodes. Their scheme ensures untraceability, mutual authentication, and session key verification. However, Wang *et al.* [20] highlighted that Amin *et al.*'s scheme is vulnerable to impersonation attacks and fails to provide user anonymity. Pirayesh *et al.* [21] presented a device authentication and key agreement scheme that utilizes hyperelliptic curve-based digital signature arithmetic (HECDSA). Their protocol achieves lower computation costs and key sizes compared to ECC. Notably, a significant portion of the computational load is offloaded to the GWN, considering the limited computation power and bandwidth of IoT devices.

3. Authentication scheme for M2M communication

Lightweight authentication protocols are specifically designed for secure machine-to-machine (M2M) communication in the context of the IIoT or vehicle-to-vehicle communication in vehicular ad-hoc networks (VANETs). These protocols are optimized to provide efficient and secure authentication mechanisms that meet the resource constraints and dynamic nature of these environments.

In highly dynamic environments like VANET, lightweight authentication schemes have been proposed to efficiently handle the authentication process. Chuang and Lee [22] introduced a decentralized lightweight authentication scheme called trust-extended authentication mechanism (TEAM). The trust extension concept allows a device to become trustworthy once authenticated by a trusted node, enabling it to authenticate other devices. However, Kumari *et al.* [23] demonstrated vulnerabilities in Chuang and Lee's scheme, including guess attacks, impersonation attacks, and insider attacks. Esfahani *et al.* [4] proposed a lightweight authentication and key agreement scheme for M2M communication in IIoT based on Chuang and Lee's scheme. However, Esfahani *et al.*'s scheme did not address the weaknesses of Chuang and Lee's scheme and remained susceptible to guess attacks and impersonation attacks.

For M2M communication of resource-constrained field devices in IoT environments, lightweight authentication and key agreement protocols have been proposed. Nakkar *et al.* [6] introduced a lightweight symmetric key-based protocol with a computational complexity of 5 hash and 4 XOR operations. The communication overhead was 171 bytes, and the storage requirement for field devices was 160 bytes. Similarly, Lara *et al.* [5] designed a lightweight authentication protocol, and Miyanaji *et al.* [7] proposed a continuous lightweight authentication according group priority and key agreement protocol. Both protocols underwent security analysis using formal methods such as the Burrows-Abadi-Needham (BAN) logic and the AVISPA tool. However, to the best of our knowledge, the majority of existing lightweight authentication and key agreement protocols for M2M communication are vulnerable to specific malicious attacks or suffer from inefficiencies in terms of computation and communication. Furthermore, most of these schemes also fail to provide untraceability. The untraceability is an important security feature because if an adversary can trace the messages of a field device, the anonymity of the device is compromised to some extent.

In this work, we propose a comprehensive authentication and key agreement framework that enables devices in ICSs to achieve mutual authentication and secure communication. Within this framework, we introduce a newly designed lightweight authentication and key agreement protocol for M2M communication that only requires two message exchanges for each authentication and ensures untraceability. Our proposed protocol demonstrates greater robustness and efficiency compared to ex-

isting lightweight protocols [4]–[7].

Note that in our system model, we refer to the user as the PLC client, the GWN as the PLC server, and the sensor node as the field device. To achieve mutual authentication and key establishment between the PLC client and the PLC server, we employ an asymmetric cryptographic key agreement protocol, while our lightweight protocol is used for the field device authentication.

III. Preliminary

In this section, we introduce two cryptography algorithms used in asymmetric cryptographic key agreement protocol. Additionally, we will present the system model and outline the security requirements in lightweight symmetric cryptographic key agreement protocol.

1. Cryptography algorithms

1) Elliptic curve cryptography

Elliptic curves are algebraic curves defined by an equation $E : y^2 = x^3 + ax + b$ over a finite field F_p, where $p > 3$ is a prime and $a, b \in F_p$. Within the context of cryptographic protocols, the key group employed is a large prime-order subgroup of the group $E(F_q)$, which comprises F_p-rational points on E. This set of rational points encompasses all solutions $(x, y) \in F_p^2$ to the curve equation, along with a special point known as the point at infinity, serving as the neutral element. The count of F_p-rational points is denoted by $\#E(F_p)$ and the prime order of the subgroup is represented as n. A fixed generator of the cyclic subgroup is commonly referred to the base point, symbolized as G. It is worth highlighting two pivotal problems associated with ECC:

- Elliptic curve discrete logarithm problem (ECDLP): Given a point $Q \in E(F_p)$, such that $P = kG$. Even if P and G are known, determining k such that $P = kG$ is computationally infeasible.
- Elliptic curve computational Diffie-Hellman problem (ECCDHP): Given two points $aG, bG \in E(F_p)$ for any $a, b \in F_p^*$, it is hard to compute $abG \in E(F_p)$.

2) SM2 digital signature algorithm

SM2 digital signature algorithm [24] is a public key cryptographic algorithm used for digital signatures and authentication. It is based on elliptic curve cryptography and offers a secure and efficient way to ensure the authenticity and integrity of digital messages.

This algorithm starts with the selection of an elliptic curve with prime order p and a base point G of order n on that curve. A private key d is generated and kept confidential. The corresponding public key $D := dG$ is derived from the private key. When creating a signature for message m, a random number (nonce) r is chosen for each signature. A temporary point $R := rG = (x, y)$ is computed on the elliptic curve. Then, the values $t := h(m) + x \pmod n$ and $s := (1 + d)^{-1}(r - td) \pmod n$ are extracted, and the values should be not equal to zero, where $h(\cdot)$ is a hash function. Verifying a signature requires the verifier to possess the public key D, the message m, and the received signature (t, s). t and s must be within specified bounds. The final point $R := sG + (t + s)D$ is calculated. If the sum of the x-coordinate of R and the output of hash function $h(m)$ equals to t, the signature is considered valid.

The security of SM2 digital signature algorithm is based on the difficulty of solving the ECDLP. The algorithm's strength lies in its ability to offer strong security with relatively small key sizes, making it suitable for resource-constrained environments.

3) Elliptic curve Diffie-Hellman ephemeral

Elliptic curve Diffie-Hellman ephemeral (ECDHE) is a key exchange protocol that leverages elliptic curve cryptography to establish secure and efficient key establishment in cryptographic protocols. It extends the classical Diffie-Hellman (DH) key exchange algorithm by introducing the concept of ephemeral key pairs. Ephemeral key pairs are generated for each key exchange session, ensuring that a fresh set of keys is used for every new session. This characteristic provides forward secrecy, meaning that even if an adversary gains access to the long-term private key of a party in the future, they cannot decipher previously exchanged session keys.

The key exchange process in the ECDHE protocol unfolds as follows: The communicating parties, commonly known as the client and server, first reach consensus on a shared elliptic curve and a generator point located on that curve. Subsequently, each party independently generates an ephemeral key pair, composed of a private key and its corresponding public key. Once these preliminary steps are complete, the client and server initiate the exchange of parameters required for generating a shared secret point. Utilizing their private keys in conjunction with the agreed-upon parameters, each party computes a shared secret point on the elliptic curve. This shared secret point serves as the input for a key derivation function (KDF) responsible for producing a shared secret key.

The security of ECDHE also relies on the computational complexity of solving the ECCDHP. In our scheme, we implemented the ECDHE using SM2 key exchange protocol [25].

2. System model

The system model is illustrated in Figure 2. Within our comprehensive authentication and key agreement framework, four types of participants are involved: the authentication server AS, PLC server, PLC client, and field device. The AS assumes the responsibility of registering other participants, while the PLC clients can access the PLC server, which controls the field devices. The PLC server acts as the bridge between the PLC clients and the field devices.

Traditionally, PLCs are embedded devices utilized for monitoring and controlling industrial processes. They are equipped with control logic programs that define the

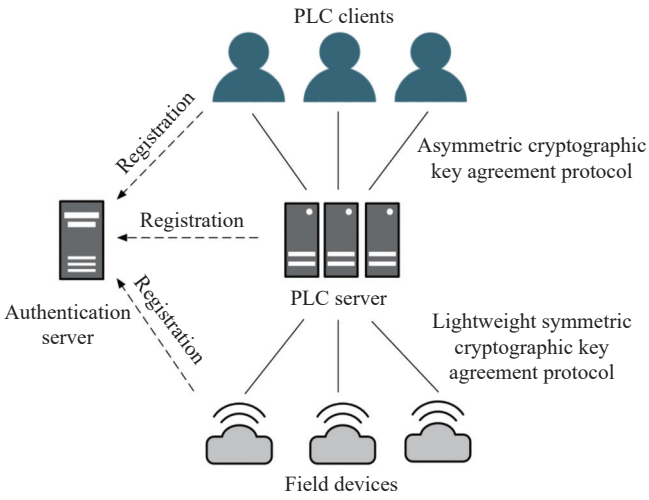

Figure 2 The system model of our general authentication and key agreement framework.

operational behavior of physical processes. These control logic programs are developed and compiled using engineering software. Conventional PLCs often feature communication interfaces (e.g., USB) on-site for interacting with engineering software. Unfortunately, these PLCs primarily rely on password-based authentication methods that are vulnerable to breaches. To address these concerns, our framework divides the PLC devices into the PLC server and the PLC clients, enabling authorized users to remotely operate the ICS over the Internet.

For authentication and key agreement between the PLC client and PLC server, we use an asymmetric cryptographic key agreement protocol that adopts the TLS handshake protocol. Considering the constrained resources of field devices, we propose a novel lightweight symmetric cryptographic key agreement protocol to ensure authentication and secure communication between the PLC server and the field devices. This lightweight protocol utilizes hash functions and XOR operations exclusively. Once the PLC clients and field devices are authenticated by the PLC server and obtain the session keys, they can securely communicate with one another. In Section III.3, we present the security requirements of the proposed lightweight protocol.

3. Security requirements

The proposed lightweight symmetric cryptographic key agreement protocol is designed to withstand various attacks, including impersonation attacks, MitM attacks, and replay attacks.

- Impersonation attack: The adversary poses as a field device or the PLC server to deceive other legitimate participants.
- MitM attack: The adversary intercepts transmitted messages, creates valid messages, and injects them into the network.
- Replay attack: The adversary repeatedly sends previously transmitted valid messages to disrupt normal communication.

In addition, our lightweight symmetric cryptographic key agreement protocol must fulfill several security requirements, as outlined in [5]–[7]. These requirements include identity anonymity, data integrity, forward secrecy, mutual authentication, and the newly added security feature of untraceability [19].

- Identity anonymity: Identity anonymity ensures that the identity numbers of field devices are not known to adversaries.
- Untraceability: Untraceability prevents adversaries from tracking field devices by eavesdropping on communication messages transmitted on the common channel.
- Integrity: Data integrity ensures that any malicious tampering of messages by adversaries is detectable.
- Forward secrecy: Forward secrecy guarantees that if the adversary obtains the current session key, previous session keys will remain secure.
- Mutual authentication: Mutual authentication requires both the PLC server and field devices to verify each other's legitimacy.

IV. Our Framework

This section provides a detailed description of the proposed framework, which integrates an asymmetric cryptographic key agreement protocol and a lightweight symmetric cryptographic key agreement protocol. We present the notations used throughout this paper in Table 1.

Table 1 Notations

Symbol	Description
λ	Security parameter
AS	Authentication server
S_i	Field device i
P_j	PLC server j
k	A secret key protected by the AS
msk	A secret key set that is pre-shared among AS and PLC servers
r_i, T_i, T_i^*	Random numbers generated by pseudo random number generator (PRNG)
id_i	The identity of entity i
aid_i	The alias of entity i
$\mathrm{sk}_{i,j}$	A session key between entity i and entity j, where $\mathrm{sk}_{i,j} = \mathrm{sk}_{j,i}$
$h(\cdot)$	A one-way hash function
$\|\|$	The concatenation operator
$\oplus$	The bitwise XOR operator

1. System overview

As illustrated in Figure 3, our framework consists of two phases: registration and authentication.

In the registration phase, the asymmetric cryptographic key agreement protocol is employed for the connection between the PLC clients and the PLC server. Specifically, we utilize the TLS handshake protocol. Ini-

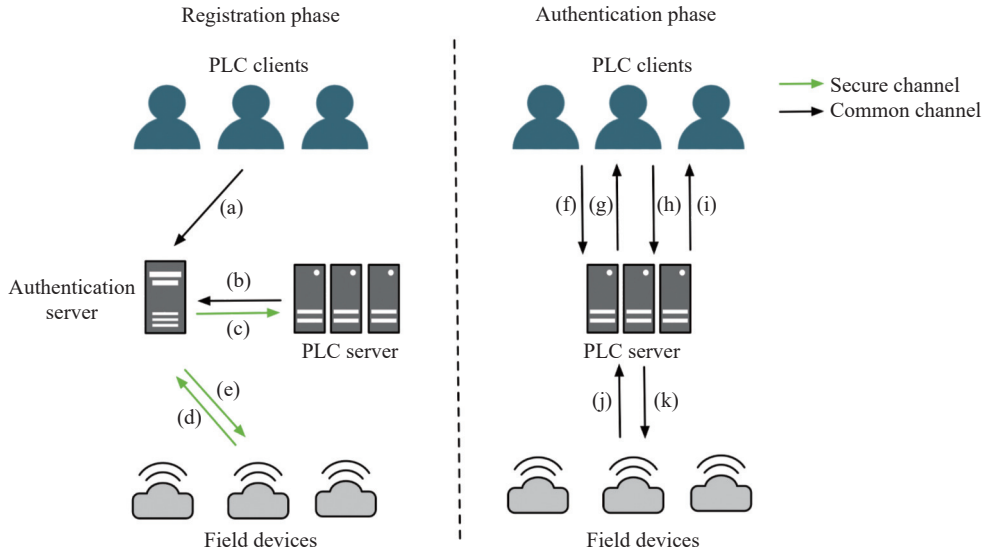

Figure 3 Overview of various phases in the proposed framework. Note that the messages (c), (d), and (e) are transmitted through secure channel. (PLC client registration phase: registration request message (a). PLC server registration phase: registration request message (b) and registration reply message (c). Field device registration phase: registration request message (d) and registration reply message (e). Asymmetric authentication and cryptographic key agreement phase: the messages (f), (g), (h), and (i) in TLS handshake. Symmetric authentication and cryptographic key agreement phase: authentication request message (j) and authentication reply message (k).).

tially, the PLC clients and the PLC server send registration request messages (a)/(b) to the authentication server AS. Upon receiving these messages, AS responds with the respective certificates, which are used to authenticate the participants' identities. For the connections between the field devices and the PLC server, a lightweight anonymous authentication and key agreement scheme is utilized. In this phase, each PLC server receives message (c) containing a pre-shared key sent by AS, while each field device sends a registration message (d) containing its identity number to AS. If the registration is successful, AS responds with message (e) containing several parameters.

In the authentication phase, the TLS handshake protocol is employed for authentication and session key agreement between the PLC client and the PLC server. Initially, the PLC client sends message (f) to the PLC server to negotiate cipher settings. The PLC server responds with message (g), which includes the specified cipher settings, the PLC server's certificate, and the DH public key. Upon receiving message (g), the PLC client verifies the PLC server's certificate and transmits its own certificate with message (h). Additionally, through message (h), the PLC client transmits the DH public key to the PLC server and notifies the PLC server that subsequent messages can be authenticated. Both the PLC server and the PLC clients compute a shared key based on a key exchange algorithm like ECDHE. Finally, the PLC server sends message (i) to inform the PLC client that subsequent transmitted messages can be authenticated. At this point, the PLC clients and the PLC server can encrypt communication messages using a symmetric cipher with the shared session key. For the industrial field devices, the field device sends an authentication request message (j). Upon receiving message (j), the PLC server calculates a session key and responds with an authentication reply message (k). The field device can then extract the session key from message (k). Subsequently, the field device and the PLC server establish a shared session key.

2. Asymmetric cryptographic key agreement protocol

The asymmetric cryptographic key agreement protocol is used for devices over the open Internet, and we adopt the TLS handshake protocol. We consider that the participants in this protocol are the PLC clients and the PLC server. In the registration phase, after receiving the registration request from a PLC client or the PLC server, AS issues the certificates to them, with which the participants can authenticate the origin of messages in the next phase. In the authentication phase, we mainly follow the TLS handshake protocol description.

A example of message transmission during the TLS handshake is depicted in Figure 4. In particular, we use SM2 digital signature algorithm [24] for signature and verification, and ECDHE implemented using SM2 key exchange protocol [25] as cryptographic key agreement algorithm.

1) Client → Server: At the outset, the PLC server and the PLC client initiate a negotiation for the TLS protocol and cipher suite. The process begins with the PLC client sending a ClientHello message to the PLC server. This message includes the protocol version, a random number R_c generated by the PLC client, and a list of supported cipher suites.

2) Server → Client: Upon receiving ClientHello, PLC server responds a ServerHello message to specify the chosen TLS protocol, a random number R_s of server,

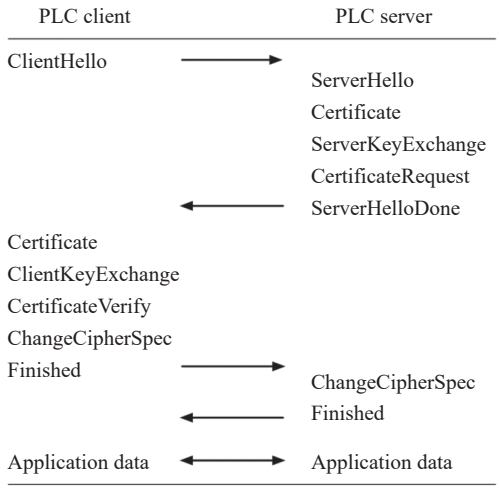

Figure 4 Message transmission during the TLS handshake.

and chosen cipher suite. Then, the PLC server transmits a Certificate message, with which the PLC client can authenticate PLC server. Next, the PLC server sends ServerKeyExchange message that contains the PLC server's the parameters in ECDHE algorithm and random numbers R_s and R_c. The PLC server uses the SM2 signature algorithm to create a signature for the parameters and the random numbers. If the server wants to verify the certificate of the client, it will send CertificateRequest to the PLC client to ask the certificate. After that, the PLC server sends ServerHelloDone message to the PLC client that indicates the handshake negotiation is done. After handshake negotiation phase, the PLC client and the PLC server exchange their random numbers R_s and R_c.

3) Client → Server: If the PLC server requires the certificate, the PLC client will send Certificate to the server. The PLC client verifies the certificate of server and sends his own certificate to the PLC server. If the certificate of server is valid, the PLC client responds ClientKeyExchange message that contains the client's parameters in ECDHE algorithm. Then, the PLC client sends CertificateVerify to prove the certificate is owned by the PLC client. After the PLC server received the ClientKeyExchange message, both PLC server and PLC client obtain R_s, R_c, and a shared key obtained by the ECDHE algorithm. Then, they compute a session key, which will be used for encryption and authentication in subsequent communication. The PLC client sends ChangeCipherSpec message to inform the PLC server that all messages sent by client from now on will be authenticated and encrypted. The Finished message, which includes a message authentication code (MAC) over the preceding handshake messages, is encrypted using the session key. Upon receiving the Finished message, the PLC server verifies its authenticity. If the authentication fails, the handshake is deemed unsuccessful.

4) Server → Client: Upon successfully verifying the Certificate message and decrypting the received Finished message from the PLC client, the PLC server proceeds to send a ChangeCipherSpec message. This message serves as a notification to the PLC client that all subsequent messages sent by the PLC server will be authenticated and encrypted. Subsequently, the PLC server transmits a Finished message to the PLC client. Upon receiving the Finished message, the PLC client follows the same procedure as the PLC server in the preceding step.

3. Lightweight symmetric cryptographic key agreement protocol

Before presenting our lightweight symmetric cryptographic key agreement protocol for the connections between field devices and PLC servers, we provide some insights into the motivation behind our design.

Intuition Esfahani *et al.* [4] proposed a lightweight authentication mechanism for M2M communication in IIoT. However, their protocol requires three message exchanges for the authentication process and is vulnerable to guess identity attacks [23], compromising identity anonymity and untraceability. In a guess identity attack, an adversary attempts to guess the identity number id of a smart device by intercepting and analyzing the exchanged messages. If one of the guesses is successful, the adversary obtains the correct id and can subsequently obtain the established session key. As a result, this protocol is susceptible to impersonation and MitM attacks. Furthermore, the protocol fails to provide untraceability, as the adversary can identify the relationship among communication messages transmitted over the common channel. To address these weaknesses, our proposed lightweight protocol introduces several enhancements. First, instead of transmitting the actual identity id, the entity uses an alias aid during the communication process. Additionally, the communication messages are masked with temporary secrets. These measures prevent the adversary from brute-forcing the correct id and hinder their ability to trace the communication messages of the smart device, ensuring identity anonymity and untraceability.

Let us denote our lightweight protocol as ExKey. At a high level, ExKey consists of four algorithms: Register, Request, Reply, and Recover. Suppose that a field device S_i and a PLC server P_j want to authenticate each other and establish a shared session key. In the registration phase, the authentication server AS executes the ExKey.Register algorithm upon receiving a unique identity number id_i for field device S_i. In the authentication phase, S_i initiates the process by executing the ExKey.Request algorithm to generate an authentication request message. Upon receiving this request, the PLC server P_j performs the ExKey.Reply algorithm, which generates the session key $\mathrm{sk}_{i,j}$ and sends a reply message to S_i. Finally, upon receiving the reply message, S_i runs the ExKey.Recover algorithm, which reveals the session key $\mathrm{sk}_{i,j}$. At this point, S_i and P_j have achieved mutual authentication and agreed upon the shared session key $\mathrm{sk}_{i,j}$. By executing the ExKey algorithms, field device S_i and PLC server P_j can securely authenticate

each other and establish a shared session key for further communication.

1) Registration phase

During the registration phase, all communications take place over a secure channel. When the PLC server registers with AS, msk is shared between AS and the PLC server via the secure channel. Similarly, when the field device registers with AS, the messages (id_i) and $(\mathrm{aid}_i, A_i, B_i, T_i)$ are also transmitted over the secure channel. As shown in Figure 5, the registration of the field device involves three steps as follows.

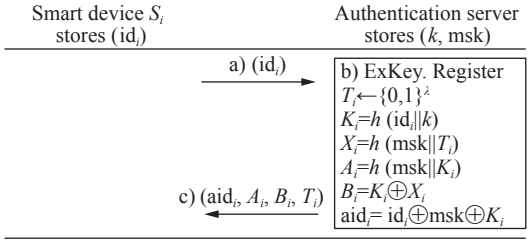

Figure 5 Registration of the field device S_i.

a) $S_i \rightarrow$ AS: The field device transmits its unique identity number id_i to AS through the secure channel.

b) ExKey.Register(id_i): Upon receiving the field device's id_i, AS executes the ExKey.Register algorithm. AS generates a random number T_i and computes the following parameters:

- $K_i = h(\mathrm{id}_i||k)$
- $X_i = h(\mathrm{msk}||T_i)$
- $A_i = h(\mathrm{msk}||K_i)$
- $B_i = K_i \oplus X_i$
- $\mathrm{aid}_i = \mathrm{id}_i \oplus \mathrm{msk} \oplus K_i$

c) AS $\rightarrow S_i$: AS transmits the parameters $(\mathrm{aid}_i, A_i, B_i, T_i)$ to the field device through the secure channel. Subsequently, AS discards these parameters.

2) Authentication

After registration phase, the field device can establish a session key with the PLC server. To ensure anonymity of the field device, the field device S_i transmits the alias aid_i instead of the real identity, and the alias is masked with a temporary secret. The authentication phase consists of the following steps, as shown in Figure 6.

a) ExKey.Request($\mathrm{aid}_i, A_i, B_i, T_i$): The field device S_i executes the ExKey.Request algorithm with parameters aid_i, A_i, and T_i. Firstly, S_i generates a random number r_i. Then, S_i computes the following values:

- $m_1 = r_i \oplus A_i$
- $R_i = h(r_i)$
- $m_2 = \mathrm{aid}_i \oplus R_i$
- $m_3 = h(\mathrm{id}_i||r_i||A_i||T_i||t_i)$

where the random number r_i is masked with the parameter A_i, and the alias aid_i is masked with the temporary secret R_i.

b) $S_i \rightarrow P_j$: The field device S_i transmits an authentication request message $(t_i, m_1, m_2, m_3, B_i, T_i)$ including timestamp t_i to the PLC server P_j. The parameters B_i and T_i allow P_j to derive r_i and id_i using the secret key msk.

c) ExKey.Reply($t_i, m_1, m_2, m_3, B_i, T_i$): Upon receiving the authentication request, the PLC server P_j executes the ExKey.Reply algorithm. P_j first check whether the timestamp t_i is valid. Then, P_j reveals r_i and id_i:

- $X_i = h(\mathrm{msk}||T_i)$
- $K_i = B_i \oplus X_i$
- $A_i = h(\mathrm{msk}||K_i)$
- $r_i = m_1 \oplus A_i$
- $R_i = h(r_i)$

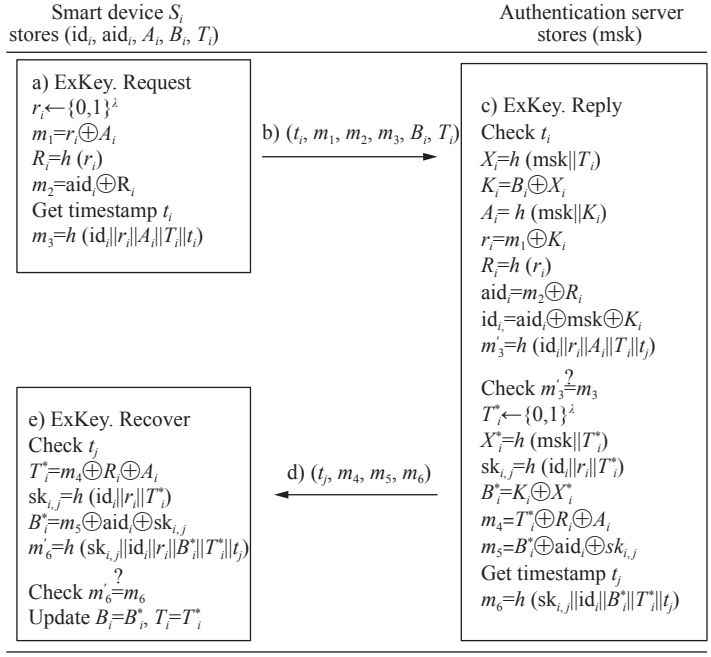

Figure 6 Authentication phase between the field device S_i and the PLC server P_j.

- $\text{aid}_i = m_2 \oplus R_i$
- $\text{id}_i = \text{aid}_i \oplus \text{msk} \oplus K_i$

P_j then checks whether $h(\text{id}_i||r_i||A_i||T_i||t_i)$ is equal to m_3. If the equality does not hold, P_j rejects the authentication request and terminates the process. Next, P_j generates a random number T_i^*. Using T_i^*, P_j calculates the session key $\text{sk}_{i,j}$ and the new parameter B_i^*:

- $X_i^* = h(\text{msk}||T_i^*)$
- $\text{sk}_{i,j} = h(\text{id}_i||r_i||T_i^*)$
- $B_i^* = K_i \oplus X_i^*$

Next, P_j computes the reply messages as follows:

- $m_4 = T_i^* \oplus R_i \oplus A_i$
- $m_5 = B_i^* \oplus \text{aid}_i \oplus \text{sk}_{i,j}$
- $m_6 = h(\text{sk}_{i,j}||\text{id}_i||B_i^*||T_i^*||t_j)$

T_i^* and B_i^* are masked with $R_i \oplus A_i$ and $\text{aid}_i \oplus \text{sk}_{i,j}$, respectively, to ensure that any two authenticated key agreement procedures cannot be linked by eavesdropping on the communication messages transmitted on the common channel.

d) $P_j \rightarrow S_i$: P_j then sends the authentication reply message (t_j, m_4, m_5, m_6) to S_i.

e) ExKey.Recover(t_j, m_4, m_5, m_6): Upon receiving the reply message, S_i executes the ExKey.Recover algorithm. If the timestamp t_j is valid, S_i obtains the new parameters T_i^* and B_i^* and recovers the session key $\text{sk}_{i,j}$ as follows:

- $T_i^* = m_4 \oplus R_i \oplus A_i$
- $\text{sk}_{i,j} = h(\text{id}_i||r_i||T_i^*)$,
- $B_i^* = m_5 \oplus \text{aid}_i \oplus \text{sk}_{i,j}$

S_i then checks whether $h(\text{sk}_{i,j}||\text{id}_i||B_i^*||T_i^*||t_j)$ is equal to m_6. If the equality holds, it confirms that P_j is trusted. Otherwise, the session key $\text{sk}_{i,j}$ is considered invalid, and S_i terminates the process. Finally, S_i updates B_i and T_i with B_i^* and T_i^*, respectively.

V. Security Analysis

This section presents a security analysis of the proposed lightweight symmetric cryptographic key agreement protocol. Hereby, we examine our protocol and demonstrate its ability to satisfy the security requirements outlined in Section III.3. Furthermore, we provide formal verification results using GNY logic and AVISPA tool to further validate the protocol's security properties. In Section V.6, we conduct a comprehensive comparison between our protocol and four other related protocols.

1. Security model

We adopt the well-known real-or-random (ROR) model [26] for key indistinguishability.

Participants $\Pi_{S_i}^u$ denotes the instance u of the field device S_i. $\Pi_{P_i}^v$ denotes the instance v of the PLC server P_i. Π_{AS}^w denotes the instance w of authentication server AS. These instances are called oracles.

Partnering Partnering is defined based on session identity (sid), which means that two instances are partnered if they hold the same sid.

Freshness We use the notation of freshness to denote that the session key between S_i and P_j is not revealed by the adversary $\mathcal{A}$.

Adversary In authentication phase, the adversary $\mathcal{A}$ can eavesdrop all transmitted messages, as well as modify the intercepted messages or forge new messages. The adversary can interact with the honest participant R through following queries, where $\Pi_{R_i}^u$ denotes the instance u of a participant R_i:

- Execute$(\Pi_{R_i}^u, \Pi_{R_j}^v)$: This query models an passive attack, in which the adversary can eavesdrop all messages during executions in authentication phase between two honest participants.
- Send$(\Pi_{R_i}^u, m)$: This query models an active attack, in which the adversary $\mathcal{A}$ can intercept a message and then send another message m to honest participant instance $\Pi_{R_i}^u$. The transmitted message m is either a modified message, or a new one, or simply the original one.
- Test$(\Pi_{R_i}^u)$: Let b be a bit value selected uniformly at random at the beginning of the experiment. If the session key is defined, the output of this query is the session key of instance $\Pi_{R_i}^u$ if $b = 1$ or a random key with the same size if $b = 0$; Otherwise, the query returns the undefined symbol $\perp$.

Semantic security in the real-or-random model The adversary $\mathcal{A}$ can ask as many Test queries as he wants to different participant instances in authentication phase. However, as the hidden bit value b was selected at the beginning, which is used to determine whether to return real key values or random key values, the outputs of Test queries are either all real session keys or all random keys. In the random case, the output of queries that are asked to two instances which are partnered should be the same random key value. In an experiment, the adversary $\mathcal{A}$'s target is to distinguish between a real session key and a random one. In the end, $\mathcal{A}$ will return a guess bit value b'. The adversary $\mathcal{A}$ is considered successful if $b = b'$.

Let Succ denote the event in which the adversary is successful. The ror-ake-advantage of $\mathcal{A}$ in breaking the semantic security of the authenticated key exchange (AKE) protocol $\mathcal{P}$ is

$$\text{Adv}_{\mathcal{P}}^{\text{ror}-\text{ake}}(\mathcal{A}) = \left|\Pr(\text{Succ}) - \frac{1}{2}\right|$$

We say that protocol $\mathcal{P}$ is a semantic security protocol in the ROR sense if $\text{Adv}_{\mathcal{P}}^{\text{ror}-\text{ake}}(\mathcal{A})$ is negligible.

Random oracle All participants as well as adversary $\mathcal{A}$ are provided with a collision-resistant one-way hash function $h()$, which is modeled by a random oracle. We use $\mathcal{H}$ to denote the random oracle.

2. Formal security analysis using ROR model

Under ROR model mentioned in Section V.1, we formally analyze the security of session key in the proposed lightweight protocol via the following theorem.

Theorem 1 Let $\mathcal{A}$ be an adversary running in poly-

nomial time t against the proposed lightweight symmetric cryptographic key agreement protocol ExKey, where the cryptographically secure hash function $h()$ is modeled as a random oracle $\mathcal{H}$. Suppose that none of the PLC servers have been compromised by the adversary. Then, we have

$$\mathsf{Adv}^{\mathsf{ror-ake}}_{\mathsf{ExKey}}(\mathcal{A}) \leq \frac{Q_h^2}{2|H|}$$

where Q_h is the numbers of $\mathcal{H}$ queries; $|H|$ is range space of the hash function $h()$; the key length is λ bits.

Proof To prove this theorem, we first define a sequence of games, denoted as G_0, G_1, and G_2. Let Succ_i denote the adversary which is successful in game G_i $(i = 0, 1, 2)$.

Game G_0**:** This game is the real attack by $\mathcal{A}$ on the proposed protocol ExKey in the random oracle model. The bit value b is selected randomly at the beginning of this game. By definition, we have

$$\mathrm{Adv}^{\mathrm{ror-ake}}_{\mathrm{ExKey}}(\mathcal{A}) = \left|\Pr(\mathrm{Succ}_0) - \frac{1}{2}\right| \tag{1}$$

Game G_1: This game simulates the passive attack by querying the Execute oracle. At the end, $\mathcal{A}$ queries the Test oracle. $\mathcal{A}$ has to decide the output of Test is the real session key $\mathrm{sk}_{i,j}$ or a random one. The session key is derived as

$$\mathrm{sk}_{i,j} = h(\mathrm{id}_i||r_i||T_i^*)$$

When $h()$ is modeled as a random oracle, the adversary needs to guess the exact input $(\mathrm{id}_i||r_i||T_i^*)$ to determine the correct value of $\mathrm{sk}_{i,j}$. Since r_i and T_i^* are temporary secrets and the adversary cannot obtain the identity id_i, the adversary cannot generate session key $\mathrm{sk}_{i,j}$. Therefore, the chance of winning by the adversary is not increased through eavesdropping the communication messages $(t_i, m_1, m_2, m_3, B_i, T_i)$ and (t_j, m_4, m_5, m_6). Then we have

$$\Pr(\mathrm{Succ}_0) = \Pr(\mathrm{Succ}_1) \tag{2}$$

Game G_2: In this game, Send and $\mathcal{H}$ oracles are added to simulate an active attack, in which $\mathcal{A}$ tries to trick a participant into accepting his forged message. $\mathcal{A}$ tries to find collisions by repeatedly querying the $\mathcal{H}$ oracle. As each value in authentication request message $(t_i, m_1, m_2, m_3, B_i, T_i)$ and authentication reply message $(t_j m_4, m_5, m_6)$ is associated to the identity of a participant, secret key values, and secret random numbers, there is no collision when querying the Send oracle. According to the birthday paradox, we have

$$|\Pr(\mathrm{Succ}_1) - \Pr(\mathrm{Succ}_2)| \leq \frac{Q_h^2}{2|H|} \tag{3}$$

In the last game, all queries have been simulated. If the adversary $\mathcal{A}$ still cannot breach the security of protocol ExKey at this point, he has to guess the bit value b to win the game after querying Test. Therefore, we have

$$\Pr(\mathrm{Succ}_2) = \frac{1}{2} \tag{4}$$

Therefore, the overall adversarial advantage is

$$\mathrm{Adv}^{\mathrm{ror-ake}}_{\mathrm{ExKey}}(\mathcal{A}) = |\Pr(\mathrm{Succ}_0) - \Pr(\mathrm{Succ}_2)| \leq \frac{Q_h^2}{2|H|} \tag{5}$$

3. Additional security properties

In this section, we demonstrate that the proposed protocol offers several important security features, including identity anonymity, untraceability, integrity, mutual authentication, and forward secrecy. Additionally, we highlight its ability to resist replay attacks, MitM attacks, and impersonation attacks.

1) Identity anonymity

Firstly, it is important to note that the plaintext identity of the field device is never transmitted over the common channel. Even if an adversary intercepts the messages exchanged during the authentication phase and obtains the values $(t_i, t_j, B_i, T_i, m_1, m_2, m_3, m_4, m_5, m_6)$, it is computationally infeasible for the adversary to derive the actual identity id_i from the equations $m_3 = h(\mathrm{id}_i||r_i||A_i||T_i||t_i)$ and $m_6 = h(\mathrm{sk}_{i,j}||\mathrm{id}_i||B_i^*||T_i^*||t_j)$. This is due to the fact that the hash function $h()$ is modeled as a random oracle. Besides, equations $\mathsf{aid}_i = \mathsf{id}_i \oplus \mathrm{msk} \oplus K_i$ and $m_2 = \mathrm{aid}_i \oplus R_i$ do not reveal id_i since the adversary $\mathcal{A}$ is unaware of R_i and aid_i.

Furthermore, even if the $\mathcal{A}$ attempts to reveal the identity through a guess attack based on the intercepted authentication request and reply messages, their efforts will be futile. The adversary can guess a value id_i' as the identity of S_i, but they cannot compute aid_i' without knowing msk and K_i. Additionally, the adversary cannot calculate $h(\mathrm{id}_i'||r_i||A_i||T_i||t_i)$ without knowing r_i and A_i, or $h(\mathrm{sk}_{i,j}||\mathrm{id}_i'||B_i^*||T_i^*||t_j)$ without knowing $\mathrm{sk}_{i,j}$ and B_i^*. Since the adversary cannot compute m_3 and m_6, they cannot verify whether their guessed value id_i' is correct based on the messages (m_3, m_6). Therefore, our protocol effectively resists guess identity attacks and provides identity anonymity for the field devices.

2) Untraceability

The identity of the device is concealed within the alias aid_i, which is masked with the temporary secret R_i. Since $R_i = h(r_i)$ and r_i is randomly chosen by the field device for each key agreement session, it is practically impossible for the adversary $\mathcal{A}$ to determine the device's identity or establish any correlation between different authentication request messages. In the authentication reply messages, the parameters B_i^* and T_i^* are transmitted over the common channel for updating the stored values of B_i and T_i in the field device. To prevent tracking, B_i^* and T_i^* are masked with $R_i \oplus A_i$ and $\mathrm{aid}_i \oplus \mathrm{sk}_{i,j}$, re-

spectively. Consequently, the adversary $\mathcal{A}$ is unable to differentiate between the communication messages $(t_i, t_j, m_1, m_2, m_3, m_4, m_5, m_6, B_i, T_i)$ and random values. Based on the aforementioned analysis, the proposed protocol ensures untraceability, as the adversary $\mathcal{A}$ cannot trace the participants through the communication messages. Therefore, the proposed protocol satisfies the requirement of untraceability.

3) Integrity

The authentication request message includes the value m_3, and the PLC server P_j verifies whether m_3 is equal to $h(\mathrm{id}_i||r_i||A_i||T_i||t_i)$. If the request message is tampered with or modified, P_j can detect the inconsistency between $h(\mathrm{id}_i||r_i||A_i||T_i||t_i)$ and m_3. Additionally, without knowledge of the secret values msk and id_i, the adversary $\mathcal{A}$ cannot generate another valid message. Similarly, the response message contains the value m_6, and the field device S_i verifies whether m_6 matches $h(\mathrm{sk}_{i,j}||\mathrm{id}_i||B_i^*||T_i^*||t_j)$. Consequently, S_i can also detect any modifications made to the response message. Therefore, the proposed protocol ensures the integrity of transmitted messages by enabling both the PLC server and the field device to detect any unauthorized modifications.

4) Forward secrecy

In our protocol, the session key $\mathrm{sk}_{i,j}$ is derived as $\mathrm{sk}_{i,j} = h(\mathrm{id}_i||r_i||T_i^*)$, where r_i and T_i^* are random numbers, and the identity id_i remains undisclosed to the adversary $\mathcal{A}$ as demonstrated in Section V.3.1). The security of previous session keys is maintained even if the adversary gains access to the current session key. This is due to the properties of the one-way hash function and the randomness of the numbers generated independently by the field device S_i and the PLC server P_j. Consequently, our protocol guarantees forward secrecy, ensuring that past session keys cannot be compromised even if the current key is compromised.

5) Resistance to replay attack

During the authentication procedure, the inclusion of random numbers r_i and T_i^* within the request and reply messages adds an extra layer of security. If the adversary $\mathcal{A}$ manages to intercept a legitimate message and attempts to replay it to the participants, the participants will reject these replayed messages. The reason for this rejection lies in the fact that the messages $(t_i, m_1, m_2, m_3, B_i, T_i)$ and (t_j, m_4, m_5, m_6) are generated utilizing the confidential and ever-changing secrets r_i and T_i^*, respectively. Therefore, any replayed messages will feature an invalid nonce, which in turn triggers immediate rejection by the participants.

To enhance the efficiency of detecting such repeated messages, we have incorporated a timestamp into the messages. This timestamp allows participants to keep a record of only a few received messages within a short timeframe. Messages that have aged beyond this designated timeframe can be safely discarded. In the unfortunate event that the adversary attempts to replay a message long after it has been discarded, the outdated timestamp renders the replayed message invalid. Consequently, our protocol can resist replay attacks.

6) Resistance to impersonation attack

To impersonate the field device S_i and send a valid request, the adversary $\mathcal{A}$ needs to know the secret values A_i and aid_i. Without this knowledge, it is impossible for $\mathcal{A}$ to calculate a valid authentication request message. Even if $\mathcal{A}$ randomly selects values A_i' and aid_i' and sends a request message to the PLC server P_j, P_j will compute r_i' or id_i' and discover that m_3 does not match $h(\mathrm{id}_i'||r_i'||A_i'||T_i||t_i)$. Moreover, as $\mathcal{A}$ cannot obtain the secret value msk from intercepted messages due to the one-way property of the hash function, it is unable to compute a valid m_3' that should equal $h(\mathrm{id}_i'||r_i'||A_i'||T_i||t_i)$. To impersonate a field device and receive a response message from the PLC server, the adversary must acquire the secret key msk to compute valid T_i^* and $\mathrm{sk}_{i,j}$. However, $\mathcal{A}$ cannot obtain the actual r_i from messages transmitted on the common channel.

Impersonating the PLC server requires access to the secret key msk. Without this key, $\mathcal{A}$ cannot retrieve the correct values of A_i and C_i. Consequently, it is impossible for $\mathcal{A}$ to compute the correct id_i and r_i, rendering them unable to generate a valid response message that satisfies $m_6 = h(\mathrm{sk}_{i,j}||\mathrm{id}_i||B_i^*||T_i^*||t_j)$. Since $h()$ is modeled as a random oracle, the field device cannot obtain the secrets msk and K_i from $A_i = h(\mathrm{msk}||K_i)$. Additionally, the field device cannot derive msk from $\mathrm{aid}_i = \mathrm{id}_i \oplus \mathrm{msk} \oplus K_i$ without K_i. Therefore, our protocol successfully resists impersonation attacks.

7) Resistance to MitM attack

During the authentication phase of our proposed protocol, the adversary $\mathcal{A}$ has the ability to intercept the transmitted messages and attempt to forge new request or reply messages. However, as explained in Section V.3.3), $\mathcal{A}$ is unable to generate valid request or reply messages. This ensures that our protocol is resilient against MitM attacks.

8) Mutual authentication

Based on the analysis provided earlier, our protocol demonstrates resistance against impersonation attacks and MitM attacks. The PLC server P_j is able to obtain the identity id_i of the field device S_i, while P_j cannot generate a valid reply message without possessing the secret key msk. As a result, both the field device and the PLC server are able to authenticate each other successfully. Therefore, our protocol ensures mutual authentication between the field device and the PLC server.

4. Formal verification with GNY logic

In line with prior research [9], [27], we employed GNY logic [28] to meticulously scrutinize the logical integrity of our protocol. GNY logic serves as a formal verification language, enabling us to ascertain that our lightweight protocol effectively fulfills its security goals. Further elaboration on GNY and the comprehensive proof can be found in Appendix A.

5. Formal verification with AVISPA

In this study, we conducted a comprehensive verification of our lightweight authentication and key agreement protocol to assess its resilience against replay attacks and MitM attacks. To accomplish this, we employed AVISPA [29], [30], an automated verification tool widely used for protocol analysis. AVISPA implements the Dolev-Yao (DY) threat model, enabling the verification of a protocol's resistance to both passive and active attacks.

Initially, we encoded our proposed protocol using the high-level protocol specification language (HLPSL). The HLPSL code encompasses three roles: the authentication server, the PLC server, and the field device, as illustrated in Figures 7–9. Furthermore, the description of communication environment and the definition of the security goal in this protocol are shown in Figure 10. Subsequently, AVISPA translated the HLPSL code into the intermediate format (IF), which was then interpreted by back-end checkers, such as on-the-fly model check (OFMC) and constraint logic-based attack searcher (CL-AtSe).

```
role authenticationserver(
  AS,P,S          : agent,
  K,Msk           : message,
  H               : hash_func,
  ASSND,ASRCV     : channel(ota))
played_by AS
def=
  local
  State           : nat,
  IDi,AIDi,Ai,Bi,Ti,Ki,Xi : message
  init State:= 1
  transition
%% Receive registration message from field device
1.State  = 1 /\ ASRCV(IDi) =|>
  State':= 4 /\ Ti':= new()
             /\ Ki':= H(IDi.K)
             /\ Xi':= H(Msk.Ti')
             /\ Ai':= H(Msk.Ki')
             /\ Bi':= xor(Ki',Xi')
             /\ AIDi':= xor(Msk,xor(IDi,Ki'))
%% Send parameters to field device
             /\ ASSND(AIDi',Ai',Bi',Ti')
end role
```

Figure 7 Role specification of the authentication server AS in HLPSL.

To evaluate the security properties of our protocol, we defined mutual authentication and session key secrecy as our primary security goals within the AVISPA simulation. In the event that participants failed to achieve mutual authentication or if the intruder successfully obtained the session key, the output generated by the back-end tools would indicate that the protocol is unsafe, along with an explanation of how the intruder exploited the protocol's vulnerabilities.

The verification results, as depicted in Figure 11, demonstrate the successful simulation of our protocol using AVISPA under the OFMC and CL-AtSe models. The results confirm that our protocol achieves the desired security goals of mutual authentication and session key secrecy, while effectively countering replay attacks and MitM attacks.

```
role plcserver(
  AS,P,S          : agent,
  Msk             : message,
  H               : hash_func,
  PSND,PRCV       : channel(dy))
played_by P
def=
  local
  State           : nat,
  IDi,AIDi,Ai,Bi,Ti,Ki,Xi,Tti,Ttn,
  M1,M2,M3,M4,M5,M6,SKij,Ri,
  Tin,Xin,Ain,Bin,M3a,M3b : message
  init State:= 2
  transition
%% Receive authentication request message from field device
1.State  = 2 /\ PRCV(Tti,M1,M2,M3,Bi,Ti) =|>
  State':= 6 /\ Xi':= H(Msk.Ti)
             /\ Ki':= xor(Bi,Xi')
             /\ Ai':= H(Msk.Ki')
             /\ Ri':= xor(M1,Ai')
             /\ AIDi':= xor(M2,H(Ri'))
             /\ IDi':= xor(AIDi',xor(Msk,Ki'))
             /\ M3a':= M3
             /\ M3b':= H(IDi'.Ri'.Ai'.Ti)
2.State  = 6 /\ M3a'=M3b' =|>
  State':= 7 /\ Tin':= new()
             /\ Ttn':= new()
             /\ Xin':= H(Msk.Tin')
             /\ SKij':= H(IDi.Ri.Tin')
             /\ secret({SKij',Tin'},sec2,{S,P})
             /\ witness(P,S,p_s_sk,SKij')
             /\ Bin':= xor(Ki,Xin')
             /\ M4':= xor(Tin',xor(H(Ri),Ai))
             /\ M5':= xor(Bin',xor(AIDi,SKij'))
             /\ M6':= H(SKij'.IDi.Bin'.Tin'.Ttn'.Ttn')
%% Send reply message to field device
             /\ PSND(Ttn',M4',M5',M6')
end role
```

Figure 8 Specification of the PLC server P_j in HLPSL.

```
role fielddevice(
  AS,P,S          : agent,
  IDi             : message,
  H               : hash_func,
  SSNDs,SRCVs     : channel(ota),
  SSND,SRCV       : channel(dy)
)
played_by S
def=
  local
  State           : nat,
  AIDi,Ai,Bi,Ti,Tti,Ttn,
  M1,M2,M3,M4,M5,M6,SKij,Ri,
  Tin,Bin,M6a,M6b : message
  init State:= 0
  transition
%% Start execution
1.State  = 0 /\ SRCV(start) =|>
%% Send registration message to AS
  State':= 3 /\ SSNDs(IDi)
%% Receive parameters from AS
2.State  = 3 /\ SRCVs(AIDi,Ai,Bi,Ti) =|>
  State':= 5 /\ Ri':= new()
             /\ Tti':= new()
             /\ M1':= xor(Ri',Ai)
             /\ M2':= xor(AIDi,H(Ri'))
             /\ M3':= H(IDi.Ri'.Ai.Ti.Tti')
             /\ secret({Ri',Ai},sec1,{P,S})
%% Send authentication request message to PLC server
             /\ SSND(Tti',M1',M2',M3',Bi,Ti)
%% Receive reply message from PLC server
3.State  = 5 /\ SRCV(Ttn,M4,M5,M6) =|>
  State':= 8 /\ Tin':= xor(M4,xor(Ri,Ai))
             /\ SKij':= H(IDi.Ri.Tin')
             /\ request(S,P,p_s_sk,SKij')
             /\ Bin':= xor(M5,xor(AIDi,SKij'))
             /\ M6a':= M6
             /\ M6b':= H(SKij'.IDi.Bin'.Tin'.Ttn)
4.State  = 8 /\ M6a'=M6b' =|>
  State':= 9 /\ Bi':= Bin
             /\ Ti':= Tin
end role
```

Figure 9 Role specification of the field device S_i in HLPSL.

6. Security comparison

Table 2 presents a comparison of the security features between our protocol and other corresponding lightweight protocols. As previously discussed, the proto-

```
role session(
  AS,P,S       : agent,
  ID,K,Msk     : message,
  H            : hash_func)
def=
  local
%% Secure channel
  ASSND,ASRCV,SSNDs,SRCVs : channel(ota),
%% Common channel
  SSND,SRCV,PSND,PRCV     : channel(dy)
%% The composition of the session
composition
  authenticationserver(AS,P,S,K,Msk,H,ASSND,ASRCV)
  /\plcserver(AS,P,S,Msk,H,PSND,PRCV)
  /\fielddevice(AS,P,S,ID,H,SSNDs,SRCVs,SSND,SRCV)
end role

role environment()
def=
  const as,p,s  : agent,
  id,k,msk      : message,
  h             : hash_func,
  p_s_sk,sec1,sec2 : protocol_id
%% Intruder knowledge
intruder_knowledge = {p,s,h}
composition
  session(as,p,s,id,k,msk,h)
end role

goal
  secrecy_of sec1,sec2
  authentication_on p_s_sk
end goal
```

Figure 10 The description of communication environment and the definition of the security goal in the protocol.

col in [4] is susceptible to guess attacks and impersonation attacks. It also lacks identity anonymity, session key secrecy, and mutual authentication. The protocols of [5] and [9] do not provide untraceability, which can compromise the privacy of participants. The protocol in [6] is vulnerable to impersonation attacks and does not achieve mutual authentication. In [7], the protocol fails to provide anonymity, untraceability, session key secrecy, and resistance against impersonation attacks. The protocol mentioned in [8] is susceptible to both replay attacks and impersonation attacks, and does not provide untraceability. In contrast, our protocol addresses the security vulnerabilities present in the protocol of [4]. It not only ensures session key secrecy, identity anonymity, untraceability, integrity, forward secrecy, and mutual authentication but also provides protection against replay attacks, impersonation attacks, and MitM attacks.

VI. Performance Comparison

In this section, we evaluate the performance of our lightweight protocol in terms of storage cost, communication overhead, and computation cost. We also compared it with the related lightweight protocols [4]–[9], and the comparison results are illustrated in Table 3. Since we use SHA256 as hash function, the lengths of $\mathrm{id}_i, \mathrm{aid}_i, \mathrm{msk}$, $\mathrm{sk}_{i,j}$, random numbers, and the outputs of hash function are all equal and are 32 bytes. The length of a timestamp is 8 bytes.

Each field device needs to store its parameters $(\mathrm{id}_i, \mathrm{aid}_i, A_i, B_i, T_i)$ and the PLC server holds the (msk). As the length of each one is 32 bytes, the storage cost of a field device is 160 bytes and the PLC server is $96m + 32$ bytes, where m is the number of receive messages within a short time frame.

In the proposed lightweight protocol, the participants transmit four messages: (id_i), $(\mathrm{aid}, A_i, B_i, T_i)$, $(t_i, m_1, m_2, m_3, B_i, T_i)$, and (t_j, m_4, m_5, m_6). In registration phase, the field device sends its identity id_i to authentication server AS and gets the response $(\mathrm{aid}, A_i, B_i, T_i)$ from the AS. In authentication phase, the field device sends its authentication request message

```
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/laka.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.03s
  visitedNodes: 2 nodes
  depth: 1 plies
```

```
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL
PROTOCOL
  /home/span/span/testsuite/results/laka.if
GOAL
  As Specified
BACKEND
  CL-AtSe
STATISTICS
  Analysed   : 0 states
  Reachable  : 0 states
  Translation: 0.00 seconds
  Computation: 0.00 seconds
```

Figure 11 AVISPA simulation results under OFMC (left) and CL-AtSe (right) models.

Table 2 Security features comparison with peer works

Ref.	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
[4]	×	×	✓	×	✓	×	×	×	×
[5]	✓	×	✓	✓	✓	✓	✓	✓	✓
[6]	✓	✓	✓	✓	✓	×	✓	×	✓
[7]	×	×	✓	✓	✓	×	✓	✓	×
[8]	✓	×	✓	✓	×	×	✓	✓	✓
[9]	✓	×	✓	✓	✓	✓	✓	✓	✓
Ours	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note: S_1: identity anonymity; S_2: untraceability; S_3: integrity; S_4: forward secrecy; S_5: replay attack; S_6: impersonation attack; S_7: MitM attack; S_8: mutual authentication; and S_9: session key secrecy.

Table 3 Comparison results in terms of storage cost, communication overhead, and computation cost of PLC server (P_j) and device (S_i), in registration stage (Reg.) and authentication stage (Auth.)

Schemes	Storage cost		Communication overhead		Computation cost		Time cost
	S_i (B)	P_j (B)	Reg. (B)	Auth. (B)	S_i (ms)	P_j (ms)	Auth. (ms)
[4]	96	32	96	384	$7t_h + 4t_x \approx 7t_h$	$7t_h + 6t_x \approx 7t_h$	0.0063
[5]	96	96	160	384	$8t_h + 3t_x \approx 8t_h$	$8t_h + 3t_x \approx 8t_h$	0.0072
[6]	64	$96n$	232	233	$6t_h + 4t_x \approx 6t_h$	$7t_h + 4t_x \approx 7t_h$	0.0059
[7]	160	$97n + 32$	96	240	$5t_h + 5t_x \approx 5t_h$	$7t_h + 10t_x \approx 7t_h$	0.0054
[8]	128	32	128	264	$6t_h + 6t_x \approx 6t_h$	$11h_t + 6h_x \approx 11h_t$	0.0077
[9]	128	$64n + 32$	160	420	$8t_h + 5t_x \approx 8t_h$	$9h_t + 5h_x \approx 9h_t$	0.0077
Ours	160	$96m + 32$	160	272	$4t_h + 6t_x \approx 4t_h$	$7t_h + 10t_x \approx 7t_h$	0.0050

Note: n = the number of field devices that connected to a PLC server. m = the number of received messages within a short timeframe. B = bytes, $t_x \approx$0.0000036 ms, $t_h \approx$ 0.00045 ms.

$(t_i, m_1, m_2, m_3, B_i, T_i)$ to the PLC server and gets the reply message (t_j, m_4, m_5, m_6) from the PLC server. Therefore, the communication cost of registration phase is 160 bytes and authentication phase is 272 bytes.

We conducted simulations to measure the time cost of the hash operation and the XOR operation using the OpenSSL library. The simulations were performed on an Intel(R) Core(TM) i7-10700 2.90 GHz machine running the 64-bit Windows 10 operating system with 32 GB of memory. Let t_h and t_x denote the time required for a hash operation and an XOR operation, respectively. Based on the simulation results, t_h was found to be 0.00045 ms, while t_x was measured at 0.0000036 ms. Considering the significantly smaller computation cost of the XOR operation compared to the hash operation, we can neglect it when estimating the overall computation time. In the registration phase, only the authentication server AS performs the ExKey.Register operation, which incurs a computation cost of $3t_h + 3t_x$. During the authentication phase, the field device S_i executes the ExKey.Request and ExKey.Recover operations, while the PLC server P_j performs the ExKey.Reply operation. The total computation cost for S_i is $4t_h + 6t_x$, while for P_j, it is $7t_h + 10t_x$. Our protocol minimizes the computation burden on the field device, which only requires four hash operations. Furthermore, we conducted runtime tests using OpenSSL, which demonstrated that the authentication process can be completed within 0.005 ms.

The evaluation results demonstrate that our protocol exhibits storage costs and communication overhead comparable to those of other lightweight protocols. As illustrated in Figure 12, our protocol incurs significantly lower communication overhead in the authentication phase. Regarding computational costs, the authentication process on field devices involves only four hash operations, surpassing other lightweight protocols. The time cost in authentication phase is depicted in Figure 13, where our protocols outperform previous works.

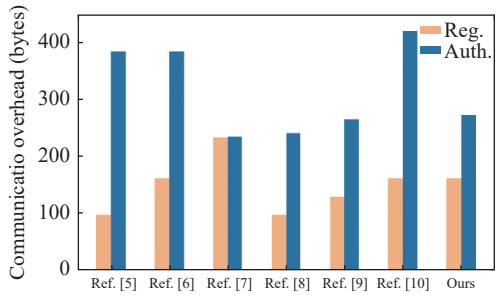

Figure 12 The communication overhead in registration and authentication phase.

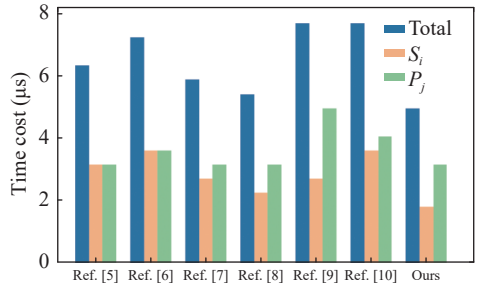

Figure 13 The time cost during authentication phase, including that of smart device (S_i), PLC server (P_j), and the total.

VII. Conclusions

In this paper, we have introduced a comprehensive authentication and key agreement framework for ICS. Our framework combines an asymmetric cryptographic key agreement protocol for devices operating over the open network and a lightweight symmetric cryptographic key agreement protocol for industrial field devices within the internal network. The general authentication framework enables secure authentication and key agreement between devices in the ICS. For devices communicating over the open network, we leverage the TLS handshake protocol to establish a secure communication channel. On the other hand, for industrial field devices within the internal network, we have developed a lightweight symmetric cryptographic key agreement protocol to achieve mutual authentication and key establishment.

The security analysis of our protocol demonstrates its effectiveness in providing essential security properties. It ensures session key secrecy, identity anonymity, untraceability, message integrity, forward secrecy, and mu-

tual authentication. Furthermore, it is resilient against impersonation attacks, MitM attacks, and replay attacks. To verify the security of our protocol, we have performed formal verification using the GNY logic and AVISPA tool, further validating its robustness and reliability. Additionally, our protocol exhibits efficiency and lightweight characteristics in terms of computation cost, communication overhead, and storage requirements. The field device only needs to perform four hash operations during each authentication process, and the total time for each authentication process is measured to be 0.005 ms in our test results. It is designed to minimize resource consumption while maintaining a high level of security for ICSs.

Overall, our proposed framework and protocol offer a strong security foundation for authentication and secure communication in ICS, ensuring the protection of critical infrastructure against potential threats and attacks.

Appendix A. Formal Verification with GNY Logic

GNY logic consists of a set of rules and postulates to reason about what principals believe about whom. Its notations are introduced in Table A-1.

Table A-1 GNY logic notation

Symbol	Description
P, Q	Principals
X, Y	Statements
(X, Y)	Conjunction
$H(X)$	One-way function
$F(X_1, ..., X_n)$	Many-to-one computationally feasible function for any X_i
$*X$	X is a not-originated-here
$P \triangleleft X$	P is told X
$P \ni X$	P possesses, or is capable of possessing X
$P \mid\sim X$	P once said X
$P \mid\equiv X$	P believes the statement X
$P \mid\equiv \sharp(X)$	P believe X is fresh, which means X has not been used for the same purpose before the current run of the protocol
$P \mid\equiv \phi(X)$	P would recognise X if P has certain expectations about the content of X before actually receiving X
$P \mid\equiv P \overset{K}{\longleftrightarrow} Q$	P believes that K is a suitable secret for P and Q

Then we give GNY logic rules. The symbol $\frac{X}{Y}$ means if X, then Y.

- Being-told rules:

(T1) $$\frac{P \triangleleft *X}{P \triangleleft X}$$

(T2) $$\frac{P \triangleleft (X, Y)}{P \triangleleft X}$$

(T3) $$\frac{P \triangleleft F(X, Y), P \ni X}{P \triangleleft Y}$$

- Possession rules:

(P1) $$\frac{P \triangleleft X}{P \ni X}$$

(P2) $$\frac{P \ni X, P \ni Y}{P \ni (X, Y), P \ni F(X, Y)}$$

(P3) $$\frac{P \ni (X, Y)}{P \ni X}$$

(P4) $$\frac{P \ni X}{P \ni H(X)}$$

(P5) $$\frac{P \ni F(X, Y), P \ni X}{P \ni Y}$$

- Freshness rules:

(F1) $$\frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X, Y), P \mid\equiv \sharp(F(X))}$$

(F2) $$\frac{P \mid\equiv \sharp(X), P \ni X}{P \mid\equiv \sharp(H(X))}$$

(F3) $$\frac{P \mid\equiv \sharp(H(X)), P \ni H(X)}{P \mid\equiv \sharp(X)}$$

- Recognizability rules:

(R1) $$\frac{P \mid\equiv \phi(X)}{P \mid\equiv \phi(X, Y), P \mid\equiv \phi(F(X))}$$

(R2) $$\frac{P \mid\equiv \phi(X), P \ni X}{P \mid\equiv \phi(H(X))}$$

(R3) $$\frac{P \ni H(X)}{P \mid\equiv \phi(X)}$$

- Message interpretation rules:

(I1) $$\frac{P \triangleleft *H(X, < S >), P \ni (X, S), P \mid\equiv P \overset{S}{\longleftrightarrow} Q, P \mid\equiv \sharp(X, S)}{P \mid\equiv Q \mid\sim (X, < S >), P \mid\equiv Q \mid\sim H(X, < S >)}$$

(I2) $$\frac{P \mid\equiv Q \mid\sim X, P \mid\equiv \sharp(X)}{P \mid\equiv Q \ni X}$$

(I3) $$\frac{P \mid\equiv Q \mid\sim (X, Y)}{P \mid\equiv Q \mid\sim X}$$

The security goals of our lightweight protocol are as follows. S_i represents the smart device, and P_j is the PLC server.

- $G_1 : S_i \mid\equiv \sharp(\mathrm{sk}_{ij})$
- $G_2 : S_i \mid\equiv \phi(\mathrm{sk}_{ij})$
- $G_3 : P_j \mid\equiv \sharp(\mathrm{sk}_{ij})$
- $G_4 : P_j \mid\equiv \phi(\mathrm{sk}_{ij})$
- $G_5 : S_i \mid\equiv P_j \ni \mathrm{sk}_{ij}$

The idealized form transmitted messages of our lightweight protocol are the following:

- $M_1 . S_i \rightarrow P_j : *F(*r_i, A_i), *F(\mathrm{id}_i, \mathrm{msk}, K, *H(*r_i)),\ *H(\mathrm{id}_i, *r, A_i, *T_i), *F(K_i, *X_i), *T_i$
- $M_2 . P_j \rightarrow S_i : *F(*T_i^*, *H(*r_i), A_i), *F(*B_i^*, \mathrm{aid}_i, *\mathrm{sk}_{ij}),\ *H(*\mathrm{sk}_{ij}, \mathrm{id}_i, *B_i^*, *T_i^*)$

The assumptions about our protocol initial state are the following:

- $A_1 : S_i \ni (\mathrm{id}_i, \mathrm{aid}_i, r_i, T_i, A_i)$
- $A_2 : P_j \ni (\mathrm{msk}, T_i^*)$
- $A_3 : S_i \mid\equiv \sharp(r_i),\ S_i \mid\equiv \phi(r_i),\ S_i \mid\equiv \sharp(T_i), S_i \mid\equiv \phi(T_i),\ S_i \mid\equiv \phi(\mathrm{id}_i), S_i \mid\equiv \phi(A_i)$
- $A_4 : P_j \mid\equiv \sharp(T_i^*), P_j \mid\equiv \phi(T_i^*), P_j \mid\equiv \phi(\mathrm{msk})$
- $A_5 : S_i \mid\equiv S_i \overset{\mathrm{id}_i}{\longleftrightarrow} P_j$

The proof then proceeds as follows: From M_1 of the idealized form of our protocol, we obtain:

- Step 1: $P_j \triangleleft *F(*r_i, A_i), *F(\mathrm{id}_i, \mathrm{msk}, K, *H(*r_i)),\ *H(\mathrm{id}_i, *r, A_i, *T_i), *F(K_i, *X_i), *T_i$. Using T_1, we get $P_j \triangleleft F(r_i, A_i), F(\mathrm{id}_i, \mathrm{msk}, K,$

$H(r_i)), H(\mathrm{id}_i, r, A_i, T_i),\ F(K_i, X_i), T_i$.

• Step 2: From Step 1, using A_2 and P_1, P_2, P_4, we get: $P_j \ni X_i$. Using A_4 and R_1, R_2, we get: $P_j \mid\equiv \phi(X_i)$.

• Step 3: From Step 2, using P_1, P_5, we get: $P_j \ni K_i$. Using R_1, we get: $P_j \mid\equiv \phi(K_i)$.

• Step 4: From Step 3, using A_2 and P_5, we get: $P_j \ni A_i$. Using R_1, R_2, we get: $P_j \mid\equiv \phi(A_i)$.

• Step 5: From Step 4, using P_5, we get: $P_j \ni r_i$. Using R_1, we get: $P_j \mid\equiv \phi(r_i)$.

• Step 6: From Step 5, using P_4, we get: $P_j \ni R_i$. Using R_2, we get: $P_j \mid\equiv \phi(R_i)$.

• Step 7: From Step 6, using A_2 and P_5, we get: $P_j \ni \mathrm{id}_i$. Using R_1, we get: $P_j \mid\equiv \phi(\mathrm{id}_i)$.

• Step 8: Using P_4, we get: $P_j \ni \mathrm{sk}_{ij}$. Using A_4 and R_2, we get: $P_j \mid\equiv \phi(\mathrm{sk}_{ij})$. (Goal G_4)

• Step 9: Using A_4 and F_1, F_2, we get: $P_j \mid\equiv \sharp\mathrm{sk}_{ij}$. (Goal G_3)

From M_2, we obtain:

• Step 10: $S_i \lhd *F(*T_i^*, *H(*r_i), A_i), *F(*B_i^*, \mathrm{aid}_i, *\mathrm{sk}_{ij}), *H(*\mathrm{sk}_{ij}, \mathrm{id}_i, *B_i^*, *T_i^*)$. Using T_1, we get $P_j \lhd F(T_i^*, R_i, A_i), F(B_i^*, \mathrm{aid}_i, \mathrm{sk}_{ij}), H(\mathrm{sk}_{ij}, \mathrm{id}_i, B_i^*, T_i^*)$.

• Step 11: Using A_1 and P_4, we get: $S_i \ni R_i$. Using A_3 and R_2, we get: $S_i \mid\equiv \phi(R_i)$.

• Step 12: From Step 11, using A_1 and P_5, we get: $S_i \ni T_i^*$. Using A_3 and R_1, we get: $S_i \mid\equiv \phi(T_i^*)$.

• Step 13: From Step 12, using P_4, we get: $S_i \ni \mathrm{sk}_{ij}$. Using A_3 and R_2, we get: $S_i \mid\equiv \phi(\mathrm{sk}_{ij})$. (Goal G_2)

• Step 14: From Step 13, using A_1 and P_5, we get: $S_i \ni B_i^*$.

• Step 15: Using A_3 and F_1, F_2, we get: $S_i \mid\equiv \sharp(\mathrm{sk}_{ij})$. (Goal G_1)

• Step 16: From Step 15, using A_7 and I_1, we get: $S_j \mid\equiv P_j \mid\sim (\mathrm{sk}_{ij}, \mathrm{id}_i, B_i^*, T_i^*)$.

• Step 17: From Step 16, using I_2, I_3, we get: $S_i \mid\equiv P_j \ni \mathrm{sk}_{ij}$. (Goal G_5)

Then the five security goals are achieved.

Acknowledgements

This work was supported by the National Key R&D Program of China (Grant No. 2021YFB3101601), the National Natural Science Foundation of China (Grant No. 62072401), and the Open Project Program of Key Laboratory of Blockchain and Cyberspace Governance of Zhejiang Province. This project is also supported by Input Output (iohk.io).

References

[1] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy Magazine*, vol. 9, no. 3, pp. 49–51, 2011.

[2] T. Tsvetanov and S. Slaria, "The effect of the colonial pipeline shutdown on gasoline prices," *Economics Letters*, vol. 209, article no. 110122, 2021.

[3] K. Stouffer, J. Falco, and K. Scarfone, "Guide to industrial control systems (ICS) security," *NIST Special Publication*, SP 800-82, 2011.

[4] A. Esfahani, G. Mantas, R. Matischek, *et al.*, "A lightweight authentication mechanism for M2M communications in industrial IoT environment," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019.

[5] E. Lara, L. Aguilar, M. A. Sanchez, *et al.*, "Lightweight authentication protocol for M2M communications of resource-constrained devices in industrial internet of things," *Sensors*, vol. 20, no. 2, article no. 501, 2020.

[6] M. Nakkar, R. AlTawy, and A. Youssef, "Lightweight authentication and key agreement protocol for edge computing applications," in *Proceedings of the IEEE 7th World Forum on Internet of Things (WF-IoT)*, New Orleans, LA, USA, pp. 415–420, 2021.

[7] R. S. Miyanaji, S. Jabbehdari, and N. Modiri, "Continuous lightweight authentication according group priority and key agreement for internet of things," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 7, article no. e4479, 2022.

[8] S. Panda, S. Mondal, and N. Kumar, "Slap: A secure and lightweight authentication protocol for machine-to-machine communication in industry 4.0," *Computers & Electrical Engineering*, vol. 98, article no. 107669, 2022.

[9] K. Shahzad, M. Alam, N. Javaid, *et al.*, "SF-LAP: Secure M2M communication in IIoT with a single-factor lightweight authentication protocol," *Journal of Sensors*, vol. 2022, article no. 1309402, 2022.

[10] K. H. M. Wong, Y. Zheng, J. N. Cao, *et al.*, "A dynamic user authentication scheme for wireless sensor networks," in *Proceedings of IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC' 06)*, Taichung, China, pp. 1–8, 2006.

[11] H. R. Tseng, R. H. Jan, and W. Yang, "An improved dynamic user authentication scheme for wireless sensor networks," in *Proceedings of IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference*, Washington, DC, USA, pp. 986–990, 2007.

[12] M. L. Das, "Two-factor user authentication in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1086–1090, 2009.

[13] M. K. Khan and K. Alghathbar, "Cryptanalysis and security improvements of 'two-factor user authentication in wireless sensor networks'," *Sensors*, vol. 10, no. 3, pp. 2450–2459, 2010.

[14] B. Vaidya, D. Makrakis, and H. T. Mouftah, "Improved two-factor user authentication in wireless sensor networks," in *Proceedings of the IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications*, Niagara Falls, ON, Canada, pp. 600–606, 2010.

[15] T. H. Chen and W. K. Shih, "A robust mutual authentication protocol for wireless sensor networks," *ETRI Journal*, vol. 32, no. 5, pp. 704–712, 2010.

[16] H. L. Yeh, T. H. Chen, P. C. Liu, *et al.*, "A secured authentication protocol for wireless sensor networks using elliptic curves cryptography," *Sensors*, vol. 11, no. 5, pp. 4767–4779, 2011.

[17] M. Turkanović, B. Brumen, and M. Hölbl, "A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the internet of things notion," *Ad Hoc Networks*, vol. 20, pp. 96–112, 2014.

[18] C. C. Chang and H. D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 1, pp. 357–366, 2016.

[19] R. Amin, S. K. H. Islam, N. Kumar, *et al.*, "An untraceable and anonymous password authentication protocol for heterogeneous wireless sensor networks," *Journal of Network and Computer Applications*, vol. 104, pp. 133–144, 2018.

[20] C. Y. Wang, D. Wang, Y. Tu, *et al.*, "Understanding node capture attacks in user authentication schemes for wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 507–523, 2022.

[21] J. Pirayesh, A. Giaretta, M. Conti, *et al.*, "A PLS-HECC-based device authentication and key agreement scheme for smart home networks," *Computer Networks*, vol. 216, article no. 109077, 2022.

[22] M. C. Chuang and J. F. Lee, "TEAM: Trust-extended authentication mechanism for vehicular ad hoc networks," *IEEE Systems Journal*, vol. 8, no. 3, pp. 749–758, 2014.

[23] S. Kumari, M. Karuppiah, X. Li, *et al.*, "An enhanced and secure trust-extended authentication mechanism for vehicular ad-hoc networks," *Security and Communication Networks*, vol. 9, no. 17, pp. 4255–4271, 2016.

[24] Standard No. GM/T 0003.2-2012:2012, Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves-Part 2: Digital Signature Algorithm, Available at: http://www.gmbz.org.cn/main/viewfile/20180108023346264349.html, 2012-03-21.

[25] Standard No. GM/T 0003.3-2012:2012, Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves-Part 3: Key Exchange Protocol, Available at: http://www.gmbz.org.cn/main/viewfile/20180108023456003485.html, 2012-03-21.

[26] M. Abdalla, P. A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proceedings of the 8th International Workshop on Public*

Key Cryptography, pp. 65–84, 2005.
[27] S. Chai, H. T. Yin, B. Xing, *et al.*, "Provably secure and lightweight authentication key agreement scheme for smart meters," *IEEE Transactions on Smart Grid*, vol. 14, no. 5, pp. 3816–3827, 2023.
[28] L. Gong, R. Needham, and R. Yahalom, "Reasoning about belief in cryptographic protocols," in *Proceedings of 1990 IEEE Computer Society Symposium on Research in Security and Privacy*, Oakland, CA, USA, pp. 234–248, 1990.
[29] A. Armando, D. Basin, Y. Boichut, *et al.*, "The AVISPA tool for the automated validation of internet security protocols and applications," in *Proceedings of the 17th International Conference on Computer Aided Verification*, Edinburgh, Scotland, UK, pp. 281–285, 2005.
[30] L. Viganò, "Automated security protocol analysis with the AVISPA tool," *Electronic Notes in Theoretical Computer Science*, vol. 155, pp. 61–86, 2006.

Shan GAO received the B.E. degree from Zhejiang University, Hangzhou, China, in 2003, the M.S. degree from the Sixth Research Institute of CEC, Beijing, China, in 2006. He is currently pursuing the Eng.D degree with the College of Engineers, Zhejiang University, Hangzhou, China. His research interest focuses on industrial information security.
(Email: higaoshan@163.com)

Junjie CHEN received the B.E. degree from Peking University, Beijing, China, in 2020. He is currently an M.S. candidate with the School of Cyber Science and Technology, Zhejiang University, Hangzhou, China. His research interests include industrial information security and data privacy.
(Email: 22121095@zju.edu.cn)

Bingsheng ZHANG received the Ph.D. degree from Worcester Polytechnic Institute, Worcester, UK. He is currently a Professor and the Associate Dean with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China, where he also directs the School of Cyber Science and Technology. Before that, he was the SUNY Empire Innovation Professor of The State University of New York at Buffalo. His current research interests include data security, IoT security, AI security, and privacy. He received Guohua Distinguished Scholar Award from Zhejiang University in 2020, IEEE CISTC Technical Recognition Award in 2017, SUNY Chancellor's Research Excellence Award in 2017, Sigma Xi Research Excellence Award in 2012, and NSF CAREER Award in 2011.
(Email: bingsheng@zju.edu.cn)

Kui REN received the B.E. degree from the Zhejiang University of Technology, Hangzhou, China, in 2007, the M.S. degree from University College London, London, UK, in 2008, and the Ph.D. degree from the University of Tartu, Tallinn, Estonia, in 2011. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. Before that, he was program director of Lancaster University's master's degree in cybersecurity and leader of the university's security research group. He specializes in cryptography, verifiable electronic voting (e-voting), and zero-knowledge proofs. In recent years, his research interests include secure computing, collaborative decision-making, and blockchain security. He has published extensively in peer-reviewed journals and conferences and received the Test-of-time Paper Award from IEEE INFOCOM and many Best Paper Awards from IEEE and ACM including MobiSys'20, ICDCS'20, Globecom'19, ASIACCS'18, ICDCS'17, etc. His h-index is 89, and his total publication citation exceeds 43000 according to Google Scholar. He is a Fellow of IEEE, a Distinguished Member of ACM and a Clarivate Highly-Cited Researcher. He is a frequent reviewer for funding agencies internationally and serves on the editorial boards of many IEEE and ACM journals. He currently serves as Chair of SIGSAC of ACM China.
(Email: kuiren@zju.edu.cn)

Xiaohua YE graduated from Zhejiang A&F University, Hangzhou, China, with a major in computer science. Now she is the CPO of Hangzhou City Brain Co., Ltd., Hangzhou, China. She has deeply involved in digtal reform of Zhejiang Province, and participated in the planning and design of the "Zhejiang Government Service Network" and "Zheliban APP". In 2018, she participated as a volunteer in the exploration and construction of Hangzhou City Brain, and has rich experience in building smart cities and digital governments. In recent years, She has led her team to independently research the digital city intensification platform CPaaS, assisting in the construction of digital governments in multiple regions within and outside Zhejiang Province. As a member of the company's digital cockpit technology team, she has helped the team win honors such as the "National Worker Pioneer" and the First Prize in the Hangzhou City Brain Digital Elite Skills Competition.
(Email: Veraye926@163.com)

Yongsheng SHEN received the Ph.D. degree in transportation planning and management from Beijing Jiaotong University, Beijing, China, and then furthered his studies as a Postdoctoral Fellow at the College of Computer Science, University of Twente, Enschede, Netherlands. He is currently the CEO of Hangzhou City Brain Co., Ltd., Hangzhou, China. Before that, he was the Technical Director of Zhejiang Daily Digital Culture Group Co., Ltd. He has published 10 high-level academic papers in domestic and foreign journals and academic conferences, including 7 papers as the first author. He has deep research and practical experience in smart city, digital governmence, smart transportation, and other related fields. The major projects he has spearheaded or participated in include Zhejiang Government Service Network, "Zheliban" APP, Hangzhou City Brain, Wenzhou City Brain, Taizhou City Brain, and more than 40 projects of digital reform in Zhejiang Province. In 2021, he led the company's technic team to win the honor of "National Worker Pioneer". Personally, he has been awarded many honorary titles, such as "Zhejiang Golden Blue Collar" and "Hangzhou May Day Labor Medal".
(Email: sys@cityos.com)