

## Article

# Privacy-Preserving ECC-Based AKA for Resource-Constrained IoT Sensor Networks with Forgotten Password Reset

Yicheng Yu <sup>1,\*</sup>, Kai Wei <sup>1</sup>, Kun Qi <sup>1,\*</sup>  and Wangyu Wu <sup>2,\*</sup> 

<sup>1</sup> School of Electronic and Communication Engineering, Shenzhen Polytechnic University, Shenzhen 518055, China

<sup>2</sup> School of Computer Science, University of Liverpool, Liverpool L69 3DR, UK

\* Correspondence: qikun@szpu.edu.cn (K.Q.); wangyu.wu@liverpool.ac.uk (W.W.)

## Abstract

Wireless sensor networks (WSNs) are extensively used in *IoT* applications. Secure access control and data protection are essential. Nonetheless, the wireless environment has an open nature. The limited resources of sensor devices render WSNs susceptible to a variety of security attacks, causing significant difficulties in the design phase of efficient authentication and key agreement (AKA) protocols. This study proposes a physically unclonable function (PUF)-based lightweight and secure AKA protocol for WSNs based on elliptic curve cryptography (ECC). A secure password update scheme is offered, which would allow legitimate users to reset forgotten passwords without re-registration. According to formal security analysis using *BAN* logic and *ProVerif*, the proposed protocol is secure against common attacks. Moreover, from an entropy perspective, the use of dynamic pseudonyms and fresh session randomness increase an adversary's uncertainty about user identities, thereby limiting identity-related information leakage. Performance evaluation shows that the proposed protocol achieves lower computational and communication overhead than the existing ones, making it suitable for WSNs with resource constraints.

**Keywords:** wireless sensor networks; authentication and key agreement; elliptic curve cryptography; physically unclonable function; forgotten password reset

## 1. Introduction

WSNs are used in many scenarios, for example, monitoring the environment, enabling healthcare-related services, supporting automated industrial processes, and improving transportation intelligence [1,2]. In this case, a large number of sensor nodes are used for collecting sensitive data and sending those data to authentic users using wireless communication channels. The open nature of wireless transmission, the unattended deployment at the sensor node's end, and the limited physical resources or capabilities of the sensor nodes make WSN a prime target of various security issues like spoofing of node identities, replay-based intrusion, communication hijacking, and offline guessing of authentication secrets, etc. [3–6].

Authentication and Key Agreement (AKA) protocols are therefore indispensable components for securing WSN communications. A well-designed AKA protocol which guarantees that only legitimate users and sensor nodes are able to access the network service and fresh session keys are established to secure the transmission of further packets. Nonetheless, the task of designing efficient and secure AKA protocols for WSNs is challenging. On the one hand, sensor nodes have limited computation capability, memory,



Academic Editors: Samsul Huda and Yasuyuki Nogami

Received: 8 January 2026

Revised: 30 January 2026

Accepted: 5 February 2026

Published: 6 February 2026

**Copyright:** © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

and battery power, which restricts the adoption of heavyweight cryptographic techniques. On the other hand, *AKA* protocols are required to provide strong security properties, which not only offer mutual authentication but also privacy preservation, perfect forward secrecy, and robustness against known attacks under powerful adversaries [7,8]. In particular, privacy in IoT-based authentication can be intuitively understood from an entropy perspective, where higher uncertainty of user identities given the observed protocol messages implies stronger resistance to tracing and identity leakage.

In recent years, several *AKA* schemes have been suggested to tackle these issues. Due to their ability to offer high security levels while using relatively small key sizes, elliptic curve cryptography (*ECC*) has become increasingly popular in recent years. This makes *ECC* suitable for *WSNs*, which often operate under severe resource constraints [3]. Moreover, hardware security mechanisms which use techniques such as *PUFs* (Physically Unclonable Functions) have been deployed to protect sensor node secrets from hardware cloning attacks [8]. The development of quantum computing has led to the emergence of post-quantum secure *AKA* protocols which can potentially offer long-term security of *WSNs* against a quantum gains adversary [4].

Figure 1 illustrates a typical *WSN* application. Three main components are involved: users, gateway nodes, and sensor nodes. Within such a framework, the sensed data are accessed by the users through gateway nodes that connect the users with the sensor nodes. Sensor nodes cooperatively perform data sensing and forwarding tasks, while the gateway node assists in authentication, access control, and key establishment. A practical *WSN* deployment basically follows this interaction pattern that also shows the importance of using secure and lightweight *AKA* protocols to protect the communications between various entities.

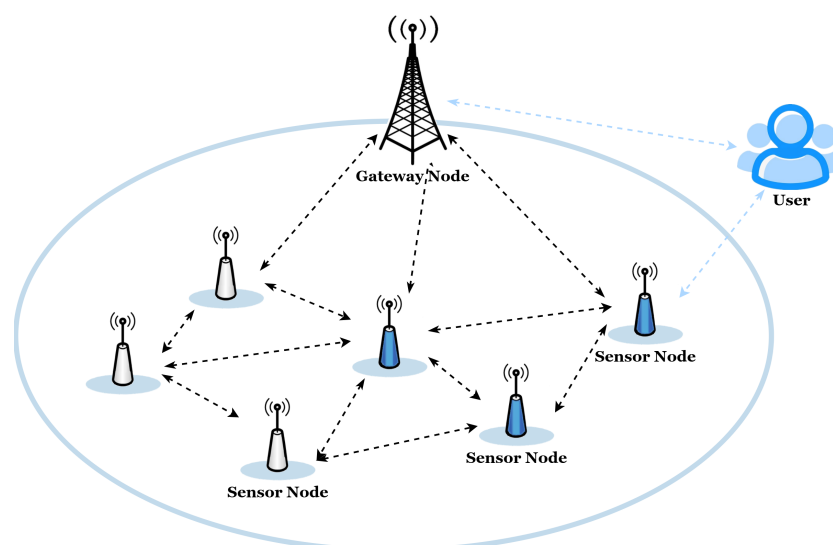


**Figure 1.** A typical *WSN* scenario.

In light of these observations, we present a secure and efficient *AKA* protocol for *WSNs* using *ECC*-based cryptographic techniques, smart-card-assisted user authentication, and *PUF*-based security of sensor nodes. We aim for strong security properties alongside lightweight computation and modest communication overhead. By combining *BAN* logic analysis, *ProVerif* security verification, and efficiency evaluation, we confirm that the proposed scheme can be applied effectively in practical *WSN* environments.

The major contributions are as follows:

- We present a lightweight *AKA* protocol for *WSNs* which achieves mutual authentication and establishes a secure session key.

- A *PUF*-based mechanism is deployed to enhance resistance of sensor nodes against physical attacks.
- The proposed protocol and its security properties are analyzed using *BAN* logic and *ProVerif*.
- We evaluate the computational and communication costs and demonstrate the efficiency of the proposed scheme compared with existing related protocols.
- We design a secure and user-friendly password update mechanism that allows legitimate users to reset forgotten passwords without requiring re-registration [9], while ensuring the overall security of the system remains intact.

## 2. Related Works

Due to various constraints such as limited computational power, energy constraints, and open wireless channels, the establishment of a secure session is very challenging. To overcome the challenges, *AKA* protocols are applied. Many lightweight and robust *AKA* schemes with strong security characteristics like mutual authentication, anonymity, forward secrecy, low computation, and communication overhead have been proposed over recent years.

A significant piece of work explores the use of Elliptic Curve Cryptography for efficient authentication. Huang et al. developed a three-factor *ECC*-based *AKA* protocol which uses biometrics, smart card, and password with formal security verification and attack resistance, which is ideally suited to resource-constrained environments [3]. To reduce overhead and improve anonymity, Li and Hu's lightweight *ECC*-based *AKA* protocol resists ephemeral secret leakage attack [7]. Many *ECC*-based designs have also been proposed which can withstand offline guessing or replay attack during the session key negotiation [10]. Simultaneously, protocol proposals for lightweight two-factor *AKA* have occurred that are based on chaotic and symmetric which are provably secure and lightweight [11].

Hybrid approaches that combine symmetric and asymmetric mechanisms to optimize energy consumption and throughput have also been explored. For instance, secure hybrid data transmission protocols integrate key management and message authentication to support efficient node authentication and data integrity [12]. Broader surveys of *IoT* security and authentication mechanisms highlight the limitations of existing *AKA* protocols and underscore the ongoing need for lightweight schemes that maintain strong security features in diverse deployment contexts [13].

Architectural innovations such as multi-gateway *AKA* schemes seek to improve scalability and flexibility in *WSNs*. Yang et al. present a multi-gateway structure that allows dynamic access for sensors and users across network regions with reduced computation and communication costs relative to earlier schemes [14]. Beyond structural enhancements, hardware-assisted authentication mechanisms using Physically Unclonable Functions (*PUFs*) have been investigated. Tyagi and Kumar introduce *PUF*-based *AKA* protocols combined with *ECC* to bolster resistance against smart card loss and physical attacks [8], and further work integrates *PUF* with chaotic maps to achieve dynamic pseudonym generation and resistance to modeling attacks [15].

The emergence of post-quantum secure *AKA* schemes is due to quantum computing threats. Singh and Mishra suggest an *AKA* protocol for resisting a quantum attacker that relies on the security of Ring Learning With Errors (RLWE) [4]. Advanced frameworks combine lightweight cryptography with context-aware key management to adapt *AKA* to both classical and post-quantum threat environments [16].

Aside from these particular designs, smart-card based authentication and fuzzy-extractor-based authentication schemes defend against card loss and impersonation attacks

with minimal overhead to sensor nodes [17]. Together, these works demonstrate the recent diversity of WSN AKA research on ECC-based multi-factor schemes, lightweight symmetric and chaotic protocols, multi-gateway structures, *PUF*-assisted schemes, and post-quantum view.

### 3. Preliminaries

#### 3.1. System Model

The system model of the proposed AKA protocol is illustrated in Figure 2. The wireless sensor network consists of three types of entities: users  $U_i$ , a gateway node  $GW$ , and multiple sensor nodes  $S_j$ .

The trusted authority is the gateway node responsible for initializing the network, registering users, and registering sensor nodes. In the registration stage, the user as well as sensor nodes submit their identity-related information to the gateway in a secure manner which provides and distributes the corresponding credentials. Once deployed, the gateway facilitates the authentication and key agreement between the users and sensor nodes but not in data transmission.

A user can access the sensed data via the gateway after performing mutual authentication and session key establishment of target sensor nodes. On successful authentication, a session key is established between the user and the sensor node for designated secure communication, as shown in Figure 2.

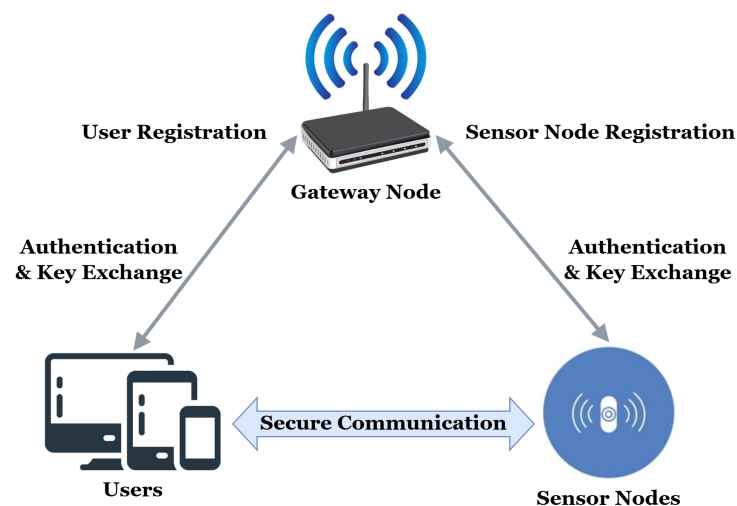


**Figure 2.** System model.

#### 3.2. Threat Model

The protocol's security analysis follows the Dolev–Yao adversarial model [18]. The adversary is assumed to have complete control of the public communication channel, where they can eavesdrop, intercept, modify, replay, and forge messages. However, the adversary cannot break standard cryptographic primitives, such as one-way hash functions and elliptic curve cryptographic operations.

Moreover, the adversary can physically take over the sensor nodes or steal user smart cards to get the stored data. Nonetheless, it is assumed that it is infeasible for malicious parties to extract sensitive secrets from cryptographic primitives, *PUFs* or solve the elliptic curve discrete logarithm problem.

### 3.3. Cryptographic Primitives

- **One-way hash function:** We adopt a cryptographic  $h(\cdot)$  to derive authentication values and maintain message integrity. It is assumed to be collision-resistant and pre-image-resistant.
- **Elliptic Curve Cryptography (ECC):** Consider an elliptic curve  $E(F_p)$  over the finite field  $F_p$ . Let  $G$  denote a cyclic additive group of prime order  $q$  generated by  $P$ . The security of the ECC-based computations is grounded on the intractability of the elliptic-curve discrete logarithm problem (ECDLP).
- **Physical Unclonable Function (PUF):** A PUF [19] is a hardware-rooted primitive that outputs device-unique, hard-to-predict responses for supplied challenges. It is utilized to protect sensor node secrets against physical attacks and cloning.

## 4. Proposed Protocol

The proposed protocol comprises six main phases: the initialization phase, the user registration phase, the sensor node registration phase, the mutual authentication phase, the password update phase, and the forgotten password reset phase. The gateway node executes the operations required in the first three phases. The mutual authentication phase is designed to achieve bidirectional identity verification between the user and the sensor node, and to negotiate a secure session key for ensuring the confidentiality and integrity of subsequent communications. If a password change is needed, the password update phase allows the password to be modified securely. In addition, for scenarios where a user forgets the password, the protocol provides a forgotten password reset phase, allowing the user to reset the password without providing the original one.

### 4.1. Initialization Phase

During the system initialization phase, the gateway node GW performs the offline setup of relevant parameters and selects security functions. GW selects an elliptic curve  $E$  defined over a finite field  $F_p$ , and chooses an additive group  $G$  on  $E$  of order  $q$ , where  $P$  denotes a generator of  $G$ . Then, GW generates a private key  $x$  and computes the corresponding public key  $X = xP$ . Finally, GW chooses a master key  $K$  and a secure cryptographic hash function  $h(\cdot) : \{0, 1\}^* \rightarrow \{0, 1\}^{256}$ . The public parameters  $E(F_p), G, P, X, h(\cdot)$  are subsequently published.

### 4.2. User Registration Phase

Before obtaining sensor-collected data, a user is required to enroll with the gateway. The registration steps are outlined below. When a user needs to access data collected by sensor nodes in the system, they must first complete registration with the gateway node. The process of the user registration phase is shown in Figure 3 and outlined below:

- **Credential Setup.** User  $U_i$  chooses an identity  $ID_i$  and a corresponding password  $PW_i$ , and generates a random number  $a_i$ .  $U_i$  then computes  $HPW_i = h(PW_i || a_i)$  and sends the parameter  $\{ID_i, HPW_i\}$  to the gateway GW over a secure channel. After obtaining  $\{ID_i, HPW_i\}$ , GW verifies whether  $ID_i$  is already registered. If it exists,  $U_i$  is requested to choose a different identity; otherwise, GW computes  $K_{GU} = h(ID_i || K)$  and  $A_i = K_{GU} \oplus HPW_i$ , stores  $ID_i$  in the user table, writes  $A_i$  into a smart card  $SC$ , and delivers  $SC$  securely to  $U_i$ .
- **Local Verification Setup.** After receiving  $SC$ ,  $U_i$  computes  $B_i = a_i \oplus h(ID_i || PW_i)$  and  $C_i = h(ID_i || HPW_i) \bmod M$ .  $M$  is chosen as a sufficiently large integer (e.g.,  $M = 2^k$ , where  $64 \leq k \leq 128$ ) to prevent efficient offline verification of guessed  $(ID_i, PW_i)$  pairs.

- Forgotten Password Reset Support.** To support scenarios where the user forgets the password, the protocol further provides a forgotten password reset initialization based on security questions. User  $U_i$  selects  $N$  pairwise coprime positive integers  $m_1, m_2, \dots, m_N$ , and  $N$  security questions  $Que_1, Que_2, \dots, Que_N$ , then provides the corresponding answers  $Ans_1, Ans_2, \dots, Ans_N$ . Subsequently, a secret  $S$  is constructed based on the Chinese Remainder Theorem such that it satisfies:  $S \equiv h(Ans_n) \bmod m_n, 1 \leq n \leq N$ . Finally,  $U_i$  computes  $PWS_i = PW_i \oplus S$  and writes the parameters  $\{B_i, C_i, PWS_i, Que_n, m_n, 1 \leq n \leq N\}$  into SC.

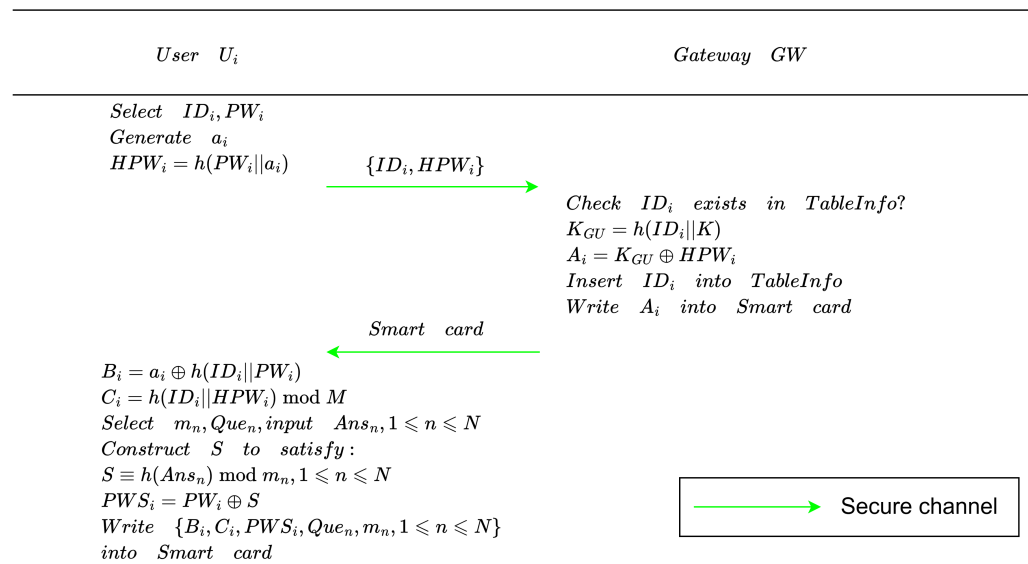


**Figure 3.** User registration phase.

#### 4.3. Sensor Node Registration Phase

Prior to deployment in the operational area, a new sensor node  $S_j$  must also complete registration with GW. The sensor registration phase binds each sensor node to the gateway and establishes a long-term credential protected by the PUF, which prevents physical cloning attacks. The sensor node registration process is shown in Figure 4, and the specific steps are described below:

- Registration request.** The sensor node  $S_j$  picks an identity  $SID_j$  and a random challenge  $Ch_j$ , then generates the corresponding response  $Re_j$  via a Physical Unclonable Function (PUF). Subsequently,  $S_j$  transmits  $SID_j$  and  $Ch_j$  to GW over a secure channel.
- Gateway credential assignment.** Upon receiving  $SID_j$  and  $Ch_j$ , GW first verifies whether the same  $SID_j$  is already present in the sensor node information table. If a duplicate is found,  $SID_j$  is notified to regenerate its identity information. Otherwise, GW computes the key  $K_{GS} = h(SID_j || K)$ , records  $SID_j$  and  $Ch_j$  in the sensor node information table, and finally sends  $K_{GS}$  back to  $S_j$  through the secure channel. With  $K_{GS}$ ,  $S_j$  computes its local key  $K_j$  and stores it securely.

#### 4.4. Mutual Authentication Phase

When a user needs to access data collected by sensor nodes, both the user and the sensor node must perform bidirectional authentication with the assistance of the gateway node. A shared session key is also negotiated to ensure the confidentiality of subsequent communications. Since the long-term credentials are established during registration, mutual authentication is completed through four message exchanges ( $Msg_1$ – $Msg_4$ ), as follows:

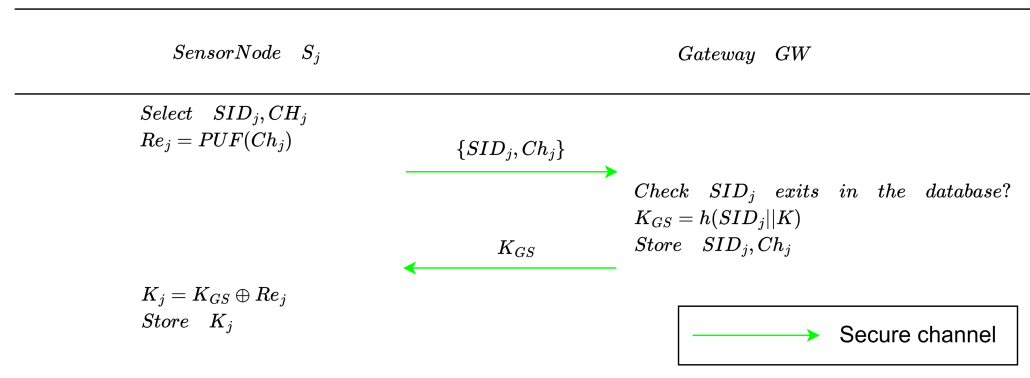


**Figure 4.** Sensor node registration phase.

- **Msg<sub>1</sub>: User ⇒ Gateway (Login request).** The user  $U_i$  inputs the identity  $ID_i$  and corresponding password  $PW_i$ , and inserts the smart card into the terminal. The terminal first performs local password verification by computing:

$$\begin{aligned} a_i^* &= B_i \oplus h(ID_i || PW_i), \\ HPW_i^* &= h(PW_i || a_i^*), \\ C_i^* &= h(ID_i || HPW_i^*) \bmod M, \end{aligned}$$

and verifies whether  $C_i^*$  matches  $C_i$ . If they do not match, the terminal rejects the login request. Otherwise, the terminal obtains the current timestamp  $T_1$ , generates random numbers  $w$  and  $r_i$ , selects the identity  $SID_j$  of the sensor node to be accessed. It successively computes:

$$\begin{aligned} K_{GU} &= A_i \oplus HPW_i, \\ D_i &= w \cdot P, E_i = w \cdot X, R_i = r_i \cdot P, \\ PID_i &= ID_i \oplus E_i, \\ M_1 &= (R_i || SID_j) \oplus K_{GU}, \\ M_{UG} &= h(ID_i || R_i || K_{GU} || M_1 || T_1). \end{aligned}$$

Finally, the terminal sends:

$$Msg_1 = \{PID_i, D_i, M_1, M_{UG}, T_1\}$$

to the gateway node  $GW$ .

- **Msg<sub>2</sub>: Gateway ⇒ Sensor (Authentication challenge).** Upon receiving  $Msg_1$ ,  $GW$  first verifies the validity of timestamp  $T_1$ , then uses its private key  $x$  to compute:

$$\begin{aligned} E_i^* &= x \cdot D_i, \\ ID_i^* &= PID_i \oplus E_i^*. \end{aligned}$$

$GW$  checks the validity of  $ID_i^*$  by querying the user information table, then computes:

$$\begin{aligned} K_{GU} &= h(ID_i^* || K), \\ (R_i^* || SID_j) &= M_1 \oplus K_{GU}, \\ M_{UG}^* &= h(ID_i^* || R_i^* || K_{GU} || M_1 || T_1). \end{aligned}$$

$GW$  compares whether  $M_{UG}^*$  equals  $M_{UG}$ . If they are not equal,  $GW$  terminates the session. Otherwise,  $GW$  generates timestamp  $T_2$ , set  $ID_i = ID_i^*$ ,  $R_i = R_i^*$ , and computes:

$$\begin{aligned} K_{GS} &= h(SID_j || K), \\ M_2 &= (ID_i || R_i) \oplus K_{GS}, \\ M_{GS} &= h(ID_i || SID_j || R_i || K_{GS} || T_2). \end{aligned}$$

GW retrieves the challenge  $Ch_j$  and sends:

$$Msg_2 = \{M_2, M_{GS}, Ch_j, T_2\}$$

to  $S_j$ .

- **Msg<sub>3</sub>: Sensor  $\Rightarrow$  Gateway (Sensor response).** After receiving  $Msg_2$ , sensor node  $S_j$  first checks the validity of timestamp  $T_2$ . Then, it generates its PUF response and computes:

$$\begin{aligned} K_{GS} &= K_j \oplus PUF(Ch_j), \\ (ID_i || R_i) &= M_2 \oplus K_{GS}, \\ M_{GS}^* &= h(ID_i || SID_j || R_i || K_{GS} || T_2). \end{aligned}$$

If  $M_{GS}^* \neq M_{GS}$  the authentication session is terminated. Otherwise,  $S_j$  generates random number  $r_j$  and computes:

$$\begin{aligned} R_j &= r_j \cdot P, \\ SK_{ji} &= h(R_i || R_j || r_j \cdot R_i), \\ M_3 &= (R_j || ID_i || SID_j) \oplus K_{GS}, \\ M_{SG} &= h(SID_j || ID_i || R_j || T_3) \end{aligned}$$

where  $T_3$  is the current timestamp and  $SK_{ji}$  is the session key. Finally,  $S_j$  returns

$$Msg_3 = \{M_3, M_{SG}, T_3\}$$

to GW.

- **Msg<sub>4</sub>: Gateway  $\Rightarrow$  User (Session confirmation).** Upon receiving  $Msg_3$ , GW checks the validity of timestamp  $T_3$  and computes:

$$\begin{aligned} (R_j || ID_i || SID_j) &= M_3 \oplus K_{GS}, \\ M_{SG}^* &= h(SID_j || ID_i || R_j || T_3). \end{aligned}$$

If  $M_{SG}^*$  matches  $M_{SG}$ , GW obtains the current timestamp  $T_4$  and computes:

$$\begin{aligned} M_4 &= (R_j || SID_j) \oplus K_{GU}, \\ M_{GU} &= h(ID_i || R_j || K_{GU} || T_4), \end{aligned}$$

and returns the message:

$$Msg_4 = \{M_4, M_{GU}, T_4\}$$

to user  $U_i$ . After receiving  $Msg_4$ ,  $U_i$  checks the validity of timestamp  $T_4$ , and computes:

$$\begin{aligned} (R_j || SID_j) &= M_4 \oplus K_{GU}, \\ M_{GU}^* &= h(ID_i || R_j || K_{GU} || T_4). \end{aligned}$$

If  $M_{GU}^*$  is verified to match  $M_{GU}$ ,  $U_i$  generates the session key:

$$SK_{ij} = h(R_i || R_j || r_i \cdot R_j),$$

thereby completing mutual authentication with sensor node  $S_j$ .

The process of the mutual authentication phase described above is illustrated in Figure 5.

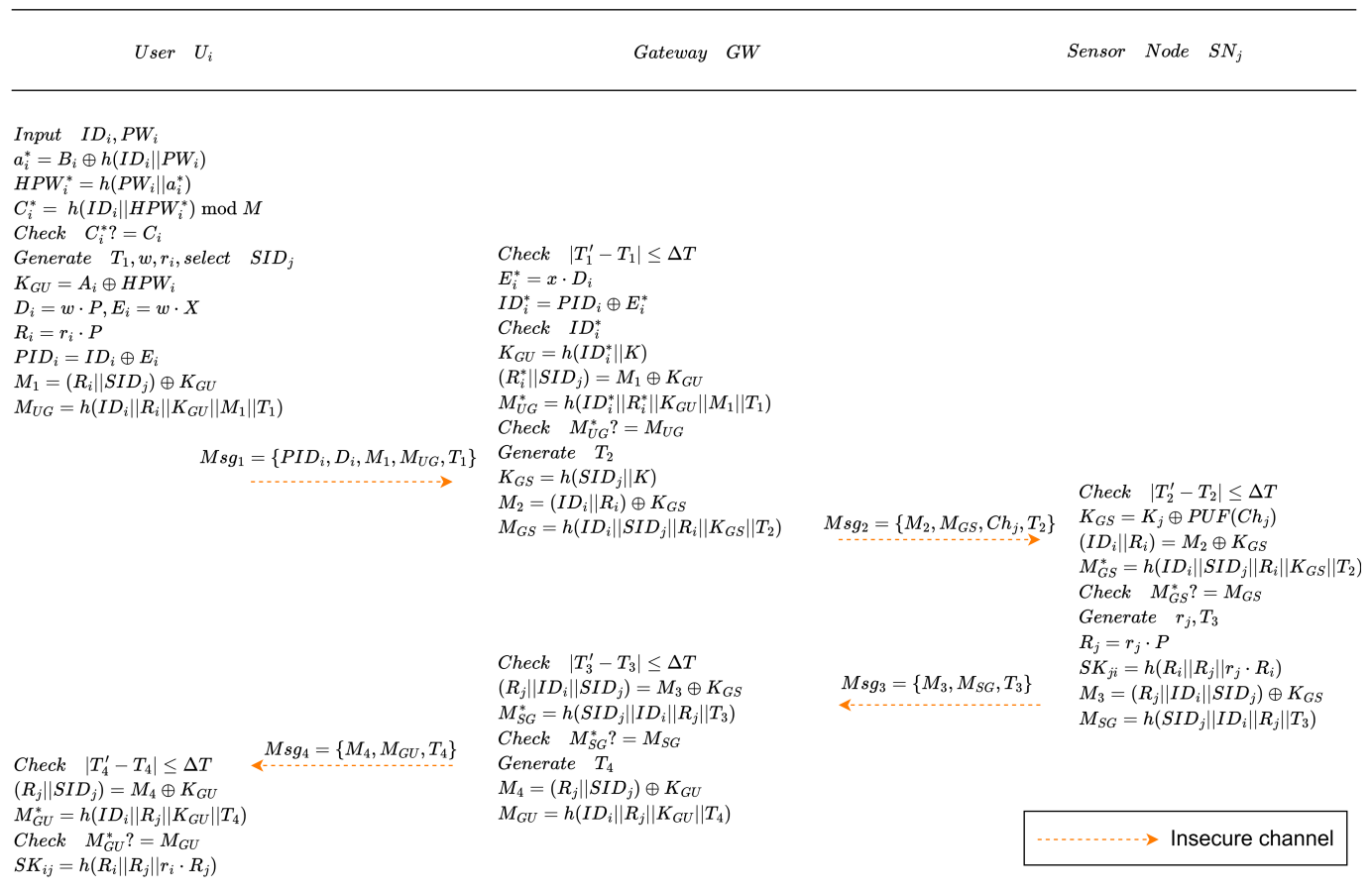


**Figure 5.** Mutual authentication phase.

#### 4.5. Password Update Phase

During the password update phase, the user can modify their password offline without interacting with the gateway node. Figure 6 is a flowchart of the process, and the specific steps are as follows:

- User  $U_i$  inserts SC into a terminal and enters the identity  $ID_i$ , the original password  $PW_i$ , and the new password  $PW_i^{new}$ .
- The terminal computes  $a_i^* = B_i \oplus h(ID_i || PW_i)$ ,  $HPW_i^* = h(PW_i || a_i^*)$ , and  $C_i^* = h(ID_i || HPW_i^*) \bmod M$ , and verifies whether the stored  $C_i$  in the smart card matches the computed  $C_i^*$ . If they do not match, the password update request is rejected. Otherwise, the terminal successively computes  $K_{GU} = A_i \oplus HPW_i^*$ ,  $HPW_i^{new} = h(PW_i^{new} || a_i^*)$ ,  $A_i^{new} = K_{GU} \oplus HPW_i^{new}$ ,  $B_i^{new} = a_i^* \oplus h(ID_i || PW_i^{new})$ ,  $C_i^{new} = h(ID_i || HPW_i^{new}) \bmod M$ , and  $PWS_i^{new} = PWS_i \oplus PW_i \oplus PW_i^{new}$ , and updates the original parameters  $A_i$ ,  $B_i$ ,  $C_i$ , and  $PWS_i$  in the smart card to  $A_i^{new}$ ,  $B_i^{new}$ ,  $C_i^{new}$  and  $PWS_i^{new}$ , respectively.

#### 4.6. Forgotten Password Reset Phase

When a user forgets their password and needs to log in, they can securely restore access through the following procedure, as shown in Figure 6.

- The user  $U_i$  inserts SC into a terminal, enters the correct identity  $ID_i$ , and provides accurate answers  $Ans_n$  to all preset security questions  $Que_n$ ,  $1 \leq n \leq N$ , thereby recovering the secret value  $S$  based on the Chinese Remainder Theorem (CRT).
- Subsequently,  $U_i$  computes the original password  $PW_i = PWS_i \oplus S$  and the parameter  $a_i = B_i \oplus h(ID_i || PW_i)$ , inputs the new password  $PW_i^{new}$ , and successively

calculates  $A_i^{new} = A_i \oplus h(PW_i || a_i) \oplus h(PW_i^{new} || a_i)$ ,  $B_i^{new} = a_i \oplus h(ID_i || PW_i^{new})$ ,  $C_i^{new} = h(ID_i || h(PW_i^{new} || a_i)) \bmod M$ , and  $PWS_i^{new} = PWS_i \oplus PW_i \oplus PW_i^{new}$ . Finally, the stored values  $A_i$ ,  $B_i$ ,  $C_i$  and  $PWS_i$  in SC are updated to the newly computed  $A_i^{new}$ ,  $B_i^{new}$ ,  $C_i^{new}$ , and  $PWS_i^{new}$ , respectively, thereby completing the secure reset of the forgotten password to  $PW_i^{new}$ .

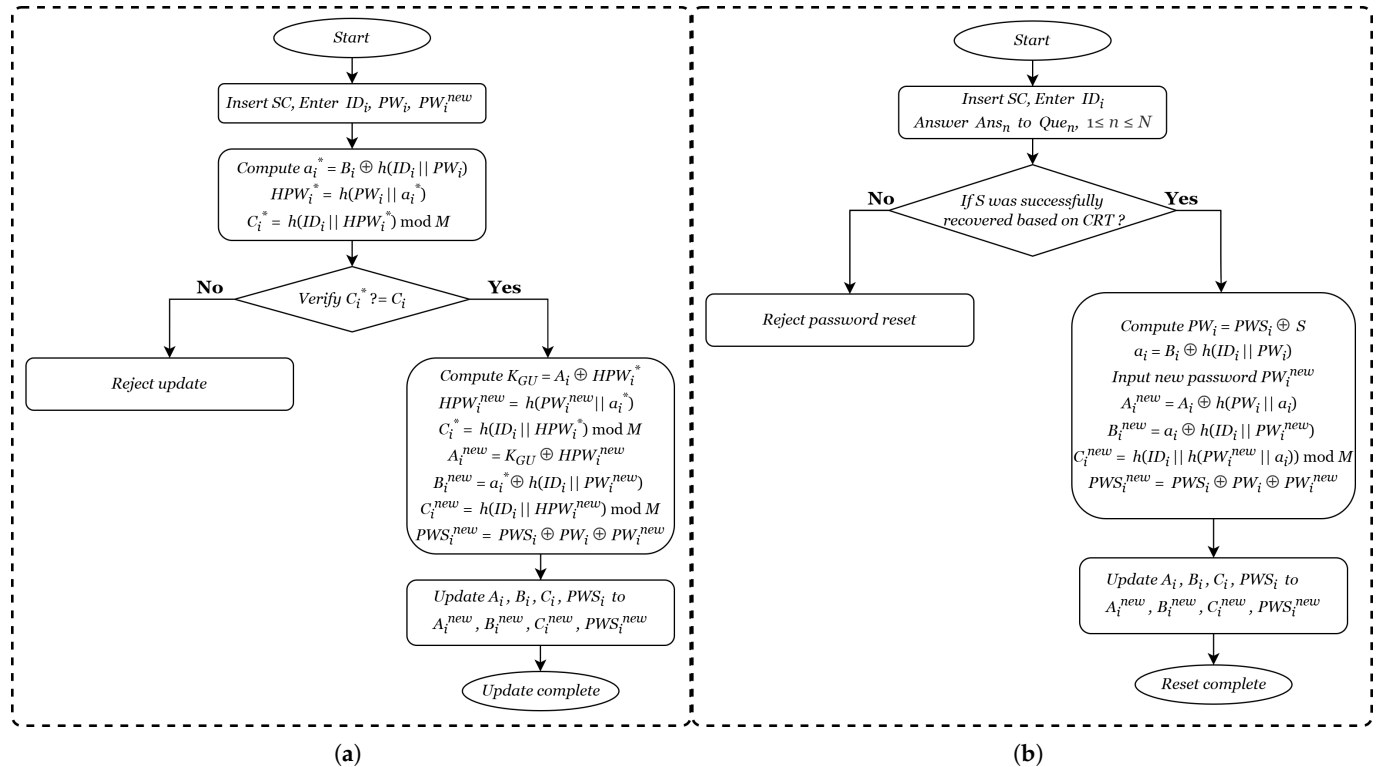


Figure 6. (a) Password update phase. (b) Forgotten password reset phase.

## 5. Security Analysis

### 5.1. Formal Security Analysis

We evaluate the proposed AKA protocol through two widely used formal methods: BAN logic and *ProVerif*. Using BAN logic, we prove that mutual authentication holds and that a session key is successfully set up. Essential security requirements (e.g., session-key confidentiality and authentication) are then machine-checked in *ProVerif* under the Dolev–Yao adversary model. The two complementary approaches provide a formal assessment of the protocol's security that is comprehensive and rigorous.

#### 5.1.1. BAN Logic-Based Security Analysis

BAN logic is a belief-based reasoning framework that is commonly adopted to analyze authentication protocols [20]. A set of inference rules abstract actual protocol messages into their idealized form and derive the principals' beliefs. For convenience, the notation set and the main inference rules are listed in Table 1 and Table 2 separately. Likewise, the idealized messages, assumptions, and step-by-step derivations allowing the authentication goal to be realized are summarized in Table 3. This reasoning shows that  $U_i$  and  $S_j$  each accept a newly generated session key and also accept that the other party accepts it, which implies mutual authentication and a secure key-agreement outcome.

**Table 1.** Basic notations of BAN logic.

| Notation                           | Meaning                                                 |
|------------------------------------|---------------------------------------------------------|
| $P \models X$                      | Principal $P$ considers $X$ to be true                  |
| $P \triangleleft X$                | Principal $P$ has received message $X$                  |
| $P \sim X$                         | At some earlier time, Principal $P$ uttered message $X$ |
| $P \Rightarrow X$                  | Principal $P$ has jurisdiction over statement $X$       |
| $\#(X)$                            | Message $X$ is fresh (i.e., has not been sent before)   |
| $P \overset{K}{\leftrightarrow} Q$ | Principals $P$ and $Q$ share a valid secret key $K$     |
| $\{X\}_K$                          | Message $X$ is encrypted using key $K$                  |
| $(X, Y)$                           | The concatenation of messages $X$ and $Y$               |

**Table 2.** Common inference rules of BAN logic.

| Rule                          | Formal Expression                                                                                      | Explanation                                                                                                              |
|-------------------------------|--------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Message Meaning Rule (MMR)    | $\frac{P \models P \overset{K}{\leftrightarrow} Q, \quad P \triangleleft \{X\}_K}{P \models Q \sim X}$ | If $P$ trusts $K$ as a shared secret with $Q$ and observes $\{X\}_K$ , then $P$ concludes that $Q$ previously sent $X$ . |
| Nonce Verification Rule (NVR) | $\frac{P \models \#(X), \quad P \models Q \sim X}{P \models Q \models X}$                              | When $P$ regards $X$ as fresh and also believes $Q$ once stated $X$ , $P$ can infer that $Q$ now believes $X$ .          |
| Jurisdiction Rule (JR)        | $\frac{P \models Q \Rightarrow X, \quad P \models Q \models X}{P \models X}$                           | If $P$ accepts $Q$ as an authority on $X$ and believes that $Q$ believes $X$ , then $P$ adopts $X$ as well.              |
| Freshness Rule (FR)           | $\frac{P \models \#(X)}{P \models \#(X, Y)}$                                                           | Freshness of $X$ implies freshness of the composite $(X, Y)$ .                                                           |
| Belief Rule (BR)              | $\frac{P \models Q \models (X, Y)}{P \models Q \models X}$                                             | If $P$ thinks $Q$ believes $(X, Y)$ , then $P$ can safely infer that $Q$ believes $X$ .                                  |

**Table 3.** The reasoning process based on BAN logic.

| Goals         |                                                                                                                                      |          |                                                                                      |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------|----------|--------------------------------------------------------------------------------------|
| $G_1$         | $U_i \models (U_i \overset{SK}{\leftrightarrow} S_j)$                                                                                | $G_2$    | $U_i \models S_j \models (U_i \overset{SK}{\leftrightarrow} S_j)$                    |
| $G_3$         | $S_j \models (U_i \overset{SK}{\leftrightarrow} S_j)$                                                                                | $G_4$    | $S_j \models U_i \models (U_i \overset{SK}{\leftrightarrow} S_j)$                    |
| Idealizations |                                                                                                                                      |          |                                                                                      |
| $Msg_1$       | $U_i \rightarrow GW : \{(R_i, SID_j), T_1\}_{KGU}$                                                                                   |          |                                                                                      |
| $Msg_2$       | $GW \rightarrow S_j : \{(U_i \overset{SK}{\leftrightarrow} S_j), (U_i \models (U_i \overset{SK}{\leftrightarrow} S_j)), T_2\}_{KGS}$ |          |                                                                                      |
| $Msg_3$       | $S_j \rightarrow GW : \{(R_j, ID_i, SID_j), T_3\}_{KGS}$                                                                             |          |                                                                                      |
| $Msg_4$       | $GW \rightarrow U_i : \{(U_i \overset{SK}{\leftrightarrow} S_j), (S_j \models (U_i \overset{SK}{\leftrightarrow} S_j)), T_4\}_{KGU}$ |          |                                                                                      |
| Assumptions   |                                                                                                                                      |          |                                                                                      |
| $A_1$         | $U_i \models (U_i \overset{KGU}{\leftrightarrow} GW)$                                                                                | $A_2$    | $GW \models (U_i \overset{KGU}{\leftrightarrow} GW)$                                 |
| $A_3$         | $S_j \models (S_j \overset{KGS}{\leftrightarrow} GW)$                                                                                | $A_4$    | $GW \models (S_j \overset{KGS}{\leftrightarrow} GW)$                                 |
| $A_5$         | $U_i \models \#(T_4)$                                                                                                                | $A_6$    | $S_j \models \#(T_2)$                                                                |
| $A_7$         | $U_i \models (GW \Rightarrow (U_i \overset{SK}{\leftrightarrow} S_j))$                                                               | $A_8$    | $S_j \models (GW \Rightarrow (U_i \overset{SK}{\leftrightarrow} S_j))$               |
| $A_9$         | $U_i \models (GW \Rightarrow (S_j \models (U_i \overset{SK}{\leftrightarrow} S_j)))$                                                 | $A_{10}$ | $S_j \models (GW \Rightarrow (U_i \models (U_i \overset{SK}{\leftrightarrow} S_j)))$ |

Table 3. Cont.

| Derivation |                                             |                                                                                                                               |
|------------|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| $D_1$      | From $Msg_4$ and $A_1$ , by MMR:            | $U_i \models GW \sim (T_4, (U_i \stackrel{SK}{\leftrightarrow} S_j), (S_j \models (U_i \stackrel{SK}{\leftrightarrow} S_j)))$ |
| $D_2$      | From $A_5$ , by FR:                         | $U_i \models \#(T_4, (U_i \stackrel{SK}{\leftrightarrow} S_j), (S_j \models (U_i \stackrel{SK}{\leftrightarrow} S_j)))$       |
| $D_3$      | From $D_1$ and $D_2$ , by NVR:              | $U_i \models GW \models ((U_i \stackrel{SK}{\leftrightarrow} S_j), (S_j \models (U_i \stackrel{SK}{\leftrightarrow} S_j)))$   |
| $D_4$      | From $D_3$ , by BR:                         | $U_i \models GW \models (U_i \stackrel{SK}{\leftrightarrow} S_j)$                                                             |
| $D_5$      | From $D_4$ and $A_7$ , by JR:               | $U_i \models (U_i \stackrel{SK}{\leftrightarrow} S_j) \quad (\text{Goal } G_1)$                                               |
| $D_6$      | From $D_3$ , by BR:                         | $U_i \models GW \models (S_j \models (U_i \stackrel{SK}{\leftrightarrow} S_j))$                                               |
| $D_7$      | From $D_6$ and $A_9$ , by JR:               | $U_i \models S_j \models (U_i \stackrel{SK}{\leftrightarrow} S_j) \quad (\text{Goal } G_2)$                                   |
| $D_8$      | From $Msg_2$ and $A_3$ , by MMR:            | $S_j \models GW \sim (T_2, (U_i \stackrel{SK}{\leftrightarrow} S_j), (U_i \models (U_i \stackrel{SK}{\leftrightarrow} S_j)))$ |
| $D_9$      | From $A_6$ , by FR:                         | $S_j \models \#(T_2, (U_i \stackrel{SK}{\leftrightarrow} S_j), (U_i \models (U_i \stackrel{SK}{\leftrightarrow} S_j)))$       |
| $D_{10}$   | From $D_8$ and $D_9$ , by NVR:              | $S_j \models GW \models ((U_i \stackrel{SK}{\leftrightarrow} S_j), (U_i \models (U_i \stackrel{SK}{\leftrightarrow} S_j)))$   |
| $D_{11}$   | From $D_{10}$ , by BR, and $A_8$ JR:        | $S_j \models (U_i \stackrel{SK}{\leftrightarrow} S_j) \quad (\text{Goal } G_3)$                                               |
| $D_{12}$   | From $D_{10}$ , by BR and $A_{10}$ , by JR: | $S_j \models U_i \models (U_i \stackrel{SK}{\leftrightarrow} S_j) \quad (\text{Goal } G_4)$                                   |

### 5.1.2. ProVerif-Based Formal Verification

To strengthen the security claims, we employ *ProVerif* [21] to perform automated formal verification under the Dolev–Yao adversary model, where the adversary can fully manipulate the public channel. The *ProVerif* code of the protocol is publicly available at [22].

In the *ProVerif* model, the user, gateway, and sensor node are specified as concurrent processes communicating over a public channel. Fresh nonces and session-related parameters are generated using *new*, while cryptographic operations are abstracted as symbolic functions. To model authentication behavior, event statements are inserted at key protocol stages, where  $UGbegin(t)$ ,  $UGend(t)$ ,  $GUBegin(t)$ , and  $GUend(t)$  denote the initiation and completion of the user–gateway authentication, and  $GSbegin(t)$ ,  $GSend(t)$ ,  $SGbegin(t)$ , and  $SGend(t)$  represent the corresponding gateway–sensor interactions. These events enable *ProVerif* to reason about agreement between protocol participants.

Authentication is verified using injective correspondence queries, such as *query inj-event(UGend(t)) ==> inj-event(UGbegin(t))*, which assert that whenever a party completes a session, there exists a unique matching session initiation by the peer, thereby providing strong resistance against replay and impersonation attacks. In addition, secrecy properties are specified using confidentiality queries of the form *query not attacker(x)*.

As shown in Figure 7, all authentication and secrecy queries are successfully verified by *ProVerif*. Specifically, the results confirm injective authentication between the user, gateway, and sensor node in all protocol phases, as each end event is associated with a unique corresponding begin event. Moreover, the secrecy queries *not attacker(svalueA[])* and *not attacker(svalueB[])* are satisfied, indicating that the attacker cannot derive the modeled sensitive values from protocol executions. Therefore, the *ProVerif* verification results in Figure 7 provide formal evidence that the scheme achieves mutual authentication and preserves the confidentiality of critical security parameters under an active adversary model.

### 5.2. Informal Security Analysis

In this part, we present an informal security discussion for the proposed authentication scheme. The analysis demonstrates that the protocol achieves the desired security goals and withstands various well-known attacks under the Dolev–Yao adversarial model, where an adversary can observe, intercept, alter, replay, or forge public-channel messages, but is assumed unable to compromise standard cryptographic primitives.

```

-----
Verification summary:

Query inj-event(UGend(t)) ==> inj-event(UGbegin(t)) is true.
Query inj-event(GUend(t)) ==> inj-event(GUbegin(t)) is true.
Query inj-event(GSend(t)) ==> inj-event(GSbegin(t)) is true.
Query inj-event(SGend(t)) ==> inj-event(SGbegin(t)) is true.
Query not attacker(svalueA[]) is true.
Query not attacker(svalueB[]) is true.
-----

```

**Figure 7.** Formal verification results of the proposed protocol using *ProVerif*.

#### 5.2.1. User Anonymity

The actual identity  $ID_i$  of the user  $U_i$  is never transmitted in plaintext on the public channel. Instead, we set a dynamic pseudonym  $PID_i = ID_i \oplus E_i$ , where  $E_i = w \cdot X$  is a random number (dependent on the session) with the public key (system)  $X$ . Since  $w$  is freshly chosen in each authentication session, the value of  $PID_i$  changes dynamically even for the same user. An external adversary cannot derive  $E_i$  or decipher the real identity  $ID_i$  without knowledge of the private key  $x$  of  $GW$ . Thus, the proposed protocol preserves user anonymity against both passive eavesdroppers and active adversaries.

#### 5.2.2. Untraceability

The protocol achieves untraceability by making sure that all user-related authentication parameters are not the same for different sessions. In particular, the messages that are transmitted include session-specific random elliptic curve points  $R_i = r_i \cdot P$  and dynamic pseudonyms  $PID_i$  using fresh random values  $r_i$  and  $w$ . Moreover, authentication messages such as  $M_{UG} = h(ID_i || R_i || K_{GU} || M_1 || T_1)$  bind the identity-related information with timestamps and ephemeral randomness. Since these parameters are statistically independent across sessions, an adversary cannot correlate multiple executions of the protocol to trace a specific user even if all communication messages are recorded.

#### 5.2.3. Mutual Authentication

Through the chained checking of authentication messages, protected by system master key derivatives, the protocol enables mutual authentication among the user, the gateway, and the sensor node. The gateway authenticates the user by validating  $M_{UG}$ , where  $K_{GU} = h(ID_i || K)$  and  $K$  is the gateway's master key. Only the holder of  $K$  must have an ability to generate a valid  $M_{UG}$ . The sensor node validates the gateway by evaluating  $M_{GS} = h(ID_i || SID_j || R_i || K_{GS} || T_2)$ . Here,  $K_{GS} = h(SID_j || K)$ , which was securely recovered using the sensor's secret key and *PUF* response. Lastly, the user authenticates the gateway by verifying the message  $M_{GU}$ , which includes the fresh sensor-generated value  $R_j$ . As a result, strong mutual authentication will be established among all entities, which will detect any forged and/or modified authentication message.

#### 5.2.4. Session Key Agreement

Once mutual authentication completes, the user and the sensor node independently compute a shared session key. The user computes  $SK_{ij} = h(R_i || R_j || r_i \cdot R_j)$ , while the sensor computes  $SK_{ji} = h(R_i || R_j || r_j \cdot R_i)$ . Due to the properties of elliptic curve scalar multiplication,  $r_i \cdot R_j = r_j \cdot R_i$ . Since the ephemeral secrets  $r_i$  and  $r_j$  are never transmitted over the public channel, an adversary cannot derive the session key even with full access to all exchanged messages.

### 5.2.5. Perfect Forward Secrecy

The proposed protocol offers perfect forward secrecy: each session key is derived from ephemeral random values  $r_i$  and  $r_j$ , which never get reused in different protocol executions. Even if long-term secrets of the user, sensor node or gateway (i.e.,  $K$ ,  $x$ , or stored credentials) are compromised at some stage, session keys established prior are secure, as recovering those would necessitate solving the *ECDLP* to extract ephemeral secrets from the past.

### 5.2.6. Forgotten Password Reset

A secure forgotten password reset mechanism is supported by the protocol. After successfully reconstructing the secret using the pre-selected security credential, the user can reset the password. Also, a password-related verifier is not revealed over the public channel. The reset process does not recover the old password; it does not destroy long-lived secrets or previous session keys. Consequently, the suggested scheme offers a secure approach for resetting forgotten passwords.

### 5.2.7. Resistance to Impersonation Attack

An adversary generates a valid authentication value  $M_{UG}$  to impersonate a legitimate user. This requires knowing  $K_{GU}$ , along with a fresh random value. It is impossible if the gateway's master key is inaccessible. In a similar context, the impersonation of a sensor node needs the generation of valid  $M_{GS}$  and  $M_{SG}$ . These values depend on  $K_{GS}$  that can be recovered with the help of a sensor's *PUF* and its secret key. In conclusion, the protocol efficiently safeguards against impersonation attempts.

### 5.2.8. Resistance to Stolen Smart Card Attack

The adversary cannot get any other information from the stolen smart card other than some hashed and masked parameters. Credential verification must pass local verification with the correct password; it is also necessary to execute the protocol successfully to generate valid gateway-authenticated messages. This means that having a smart card by itself may not impersonate the user. Thus, it may resist stolen smart card attacks.

### 5.2.9. Resistance to Offline Password Guessing Attack

The scheme prevents offline password-guessing attacks. Even when an adversary obtains smart-card-stored data through a physical access attack or a side-channel attack, it is still not feasible to uniquely verify guessed identity–password pairs  $(ID_i, PW_i)$  offline. The local verification value  $C_i$  is obtained using the modular operator, preventing an adversary from efficiently verifying guessed identity–password pairs offline. As a result, an adversary must try online login attempts to find out if a guess pair is correct. The gateway can easily detect and restrict such online attempts through monitoring and access control mechanisms. As a consequence, the proposed scheme effectively prevents offline password guessing attacks and confines the adversary to detectable online attacks.

### 5.2.10. Resistance to Known Session-Specific Temporary Information Attack

Even if all session-specific temporary data (including  $r_i$  and  $r_j$ ) are revealed, the impact is limited to that session only. The session key  $SK = h(R_i || R_j || r_i \cdot R_j)$  is created from session-dependent values beside the long-term secrets  $K$ ,  $x$ ,  $K_j$ , which include the gateway master key, gateway private key, and sensor secret, respectively, which remain secured. Because new random values are generated with every session, one session's compromise does not compromise the security of any other session. As such, the designed protocol withstands attacks that rely on exposure of session-only temporary information.

### 5.2.11. Resistance to Replay Attack

Every authentication message contains timestamps ( $T_1, T_2, T_3, T_4$ ) and fresh random numbers. Before processing received messages, the gateway, sensor, and user validate their freshness. Any replayed messages will fail to satisfy the timestamp verification or the authentication hash check, so replay attacks are prevented.

## 6. Performance Analysis

We assess the protocol's performance in terms of computational complexity and communication overhead. Since the initialization, registration, and password update phases are executed infrequently, the performance evaluation primarily focuses on the mutual authentication phase, which represents the most critical operation in practical deployments. Furthermore, we compare our protocol with the ECC-based three-factor AKA scheme by Huang et al. [3], the industrial IoT authentication protocol by Zhao et al. [23], the anonymous signature-based scheme of Vangala et al. [24], and the privacy-controlled ECC protocol REPACA proposed by Kumar et al. [25].

### 6.1. Computational Performance Analysis

During the mutual authentication phase, three entities are involved: the user  $U_i$ , the gateway node  $GW$ , and the sensor node  $S_j$ . The dominant cryptographic operations performed by these entities include: One-way hash operation, and elliptic curve scalar point multiplication, denoted as  $T_h$  and  $T_M$ , respectively. Other operations, such as XOR, concatenation, and timestamp comparison, incur negligible computational cost compared to  $T_h$  and  $T_M$  and are therefore excluded from the analysis.

During the authentication phase, the user performs the following operations:

- Three elliptic curve point multiplications to compute  $D_i = w \cdot P$ ,  $E_i = w \cdot X$  and  $R_i = r_i \cdot P$ .
- Seven hash operations for message authentication and session key generation, including the computation of  $a_i^*$ ,  $HPW_i^*$ ,  $C_i^*$ ,  $M_1$ ,  $M_{UG}$ ,  $M_{GU}^*$ , and  $SK_{ij}$ .

Hence, the total user-side computational cost is:  $3T_M + 7T_h$ .

The gateway node is responsible for identity recovery, authentication verification, and message forwarding. During the authentication phase, it performs:

- One elliptic curve point multiplication to compute  $E_i^* = x \cdot D_i$ .
- Eight hash operations to verify message integrity and authenticity, including  $K_{GU}$ ,  $M_{UG}^*$ ,  $K_{GS}$ ,  $M_2$ ,  $M_{GS}$ ,  $M_{SG}^*$ ,  $M_4$ , and  $M_{GU}$ .

Thus, the computational cost at the gateway node is:  $T_M + 8T_h$ .

Given the limited computational capability of sensor nodes, the proposed scheme is designed to minimize their cryptographic burden. During the authentication phase, the sensor node performs:

- Two elliptic curve point multiplications to compute  $R_j = r_j \cdot P$  and the shared secret component  $r_j \cdot R_i$ .
- Three hash operations for message authentication and session key derivation.

Hence, the computational cost at the sensor node is:  $2T_M + 3T_h$ .

The computational performance evaluation in this work follows the standard analytical cost estimation methodology widely adopted in lightweight AKA studies. Specifically, the execution times of basic cryptographic primitives are taken from the benchmark results reported in Srinivas et al. [26], obtained on a platform equipped with a 2.4 GHz CPU and 4 GB RAM: the time required for symmetric encryption/decryption  $T_s \approx 8.7$  ms, the execution time of a one-way hash function  $T_h \approx 0.32$  ms, the time for ECC point multiplication

$T_M \approx 17.1$  ms, the time for ECC point addition  $T_A \approx 4.4$  ms, and the execution time of the fuzzy extractor Gen/Rep function  $T_f$ , which is assumed to be approximately equal to  $T_M$ .

In our system model, the gateway node is assumed to be a relatively resource-rich entity (e.g., an edge server or base station), whereas sensor nodes are resource-constrained devices. Although ECC operations are still required at the sensor side, the proposed protocol is designed to keep the sensor-side computational workload at a minimal level compared with existing schemes, thereby ensuring feasibility in practical WSN environments.

User–gateway–sensor interactions are simulated analytically by sequentially following the authentication message flow ( $Msg_1$ – $Msg_4$ ) and counting the dominant operations executed by each participant. The total protocol cost is obtained by multiplying the operation counts by the corresponding benchmark execution times, providing a fair and reproducible comparison with related protocols.

Table 4 summarizes the computational cost comparison for the authentication phase, contrasting our scheme with four representative related protocols. As shown in Table 4, the proposed protocol exhibits the lowest overall computational cost among the compared schemes, achieving 108.4 ms per authentication session. The computational workload distribution of the proposed protocol is balanced: the gateway performs lightweight verification and forwarding, while the sensor node avoids excessive public-key operations. Such a design is particularly suitable for practical WSN deployments, where sensor-side energy consumption and latency are critical. Overall, the comparison indicates that the proposed scheme achieves competitive efficiency while retaining strong security properties.

**Table 4.** Computational cost comparison in the authentication phase with existing schemes [3,23–25].

| Protocol            | User $U_i$                       | Gateway Node $GW$     | Sensor Node $S_j$   | Total Cost (in ms) |
|---------------------|----------------------------------|-----------------------|---------------------|--------------------|
| Zhao et al. [23]    | $2T_M + 8T_h + T_f$              | $2T_M + 8T_h$         | $2T_M + 5T_h$       | 126.4 ms           |
| Vangala et al. [24] | $5T_M + 12T_h + T_A + T_s + T_f$ | $6T_M + 12T_h + 2T_A$ | $4T_M + 9T_h + T_A$ | 310.5 ms           |
| Huang et al. [3]    | $4T_M + 17T_h + T_f$             | $2T_M + 17T_h$        | $3T_M + 8T_h$       | 180.4 ms           |
| Kumar et al. [25]   | $2T_M + 4T_h$                    | $4T_M + 9T_h$         | $2T_M + 6T_h$       | 142.9 ms           |
| Proposed            | $3T_M + 7T_h$                    | $T_M + 8T_h$          | $2T_M + 3T_h$       | 108.4 ms           |

## 6.2. Communication Performance Analysis

The communication performance is evaluated by comparing the total number of transmitted bits exchanged during the authentication phase. To ensure a fair comparison, the following assumptions are adopted for data length:

- Timestamp: 32 bits.
- Random number: 256 bits.
- Hash output: 256 bits.
- Identity (ID): 128 bits.
- Elliptic curve point: 256 bits.
- PUF challenge: 128 bits.

During the authentication phase, four rounds of message exchanges are involved:

- $Msg_1 = \{PID_i, D_i, M_1, M_{UG}, T_1\}$  from  $U_i$  to  $GW$ .
- $Msg_2 = \{M_2, M_{GS}, Ch_j, T_2\}$  from  $GW$  to  $S_j$ .
- $Msg_3 = \{M_3, M_{SG}, T_3\}$  from  $S_j$  to  $GW$ .
- $Msg_4 = \{M_4, M_{GU}, T_4\}$  from  $GW$  to  $U_i$ .

In these messages,  $M_{UG}$ ,  $M_{GS}$ ,  $M_{SG}$ , and  $M_{GU}$  denote hash values, while  $T_1$ ,  $T_2$ ,  $T_3$ , and  $T_4$  represent timestamps. In addition,  $D_i$  corresponds to a point on the elliptic curve, and  $Ch_j$  denotes the PUF challenge. The values  $PID_i$ ,  $M_1$ ,  $M_2$ ,  $M_3$ , and  $M_4$  are obtained through XOR operations, whose lengths are determined by the longer operands

involved, resulting in bit-lengths of 256 bits, 384 bits, 384 bits, 512 bits, and 384 bits, respectively. Consequently, the total communication cost of the authentication phase amounts to  $256 * 4 + 32 * 4 + 256 + 128 + 256 + 384 + 384 + 512 + 384 = 3456$  bits.

The communication cost comparison between the proposed scheme and the related schemes is summarized in Table 5. As summarized in Table 5, the proposed protocol requires four message rounds, which is consistent with the compared schemes and is generally regarded as a reasonable trade-off between security and latency in WSN scenarios. Although Kumar et al. [25] report a slightly lower communication cost (3200 bits), the proposed protocol achieves substantially lower computational cost and supports richer security functionality. Therefore, from a system-level perspective, the proposed protocol offers a favorable performance trade-off.

**Table 5.** Communication cost comparison in the authentication phase with existing schemes [3,23–25].

| Protocol            | Number of Messages | Total Communication Cost (bits) |
|---------------------|--------------------|---------------------------------|
| Zhao et al. [23]    | 4                  | 3456 bits                       |
| Vangala et al. [24] | 4                  | 5152 bits                       |
| Huang et al. [3]    | 4                  | 5504 bits                       |
| Kumar et al. [25]   | 4                  | 3200 bits                       |
| Proposed            | 4                  | 3456 bits                       |

## 7. Discussion

The proposed AKA protocol presents a flexible security and efficiency trade-off for wireless sensor networks. The security of the cryptographic techniques is based on the well-established mathematics of elliptic curves. In particular, the sensor node only requires a few elliptic curve operations and hash computations during the authentication phase, which effectively reduces energy consumption.

The storage of long-term secrets in memory is avoided and physical capture and key extraction attacks are prevented to enhance the security of the sensor nodes using Physical Unclonable Functions. Furthermore, in the context of WSNs, dynamic pseudonyms and session-dependent randomness achieve user anonymity and untraceability.

The proposed protocol enhances its features with secure password update function and forgotten password reset. This is different from existing related ones. At the same time, it keeps competitive computational and communication efficiency. The design strikes a good balance among security, usability, and efficiency.

In addition to the analytical performance evaluation, it is important to consider parameter configuration in real deployment scenarios. The gateway node may adopt the widely used elliptic curve secp256r1 to balance security and computational efficiency. Since sensor nodes only perform two ECC scalar multiplications per authentication session, the proposed protocol remains feasible for resource-constrained environments. Furthermore, the timestamp tolerance window  $\Delta T$  can be set to 2–5 s depending on network latency conditions. A 128-bit PUF challenge length is recommended to enhance resistance against modeling attacks while maintaining low storage overhead. These deployment-oriented considerations further support the applicability of the proposed scheme in practical WSN-based IoT systems.

The proposed protocol is built on ECC, whose security relies on the elliptic curve discrete logarithm problem. It is known that large-scale quantum computers running Shor's algorithm may threaten ECC-based schemes. Therefore, the current design mainly targets classical security in resource-constrained WSN environments. Nevertheless, the protocol framework is modular, and ECC-based key establishment can be replaced by post-quantum

primitives (e.g., lattice-based approaches) in future extensions. This will be considered as an important direction for long-term *IoT* security.

## 8. Conclusions

In this study, we proposed a secure and efficient *AKA* protocol for *WSNs*. The proposed protocol guarantees secure mutual authentication as well as session key establishment. It also offers strong privacy protection. This was achieved through *ECC*-based smart card user authentication and *PUF*-assisted sensor authentication.

The proposed protocol was shown to be secure against impersonation, replay, offline password guessing, and stolen smart card attack through formal verification using *BAN* logic and *ProVerif* along with informal security analysis. Performance results show that our protocol reduces computation compared with several prior schemes, while keeping communication overhead at an acceptable level.

Therefore, the protocol is well suited to practical *WSN* scenarios requiring secure, privacy-preserving, and lightweight authentication. Future work on the scheme will include more complex architectures of the network and further improvement in robustness on deployment.

Future work will extend the proposed scheme in two directions: (i) conducting simulation-based *WSN* evaluations to measure authentication delay and energy consumption under varying node densities and network scales, and (ii) integrating lightweight post-quantum key agreement primitives to improve long-term security against quantum adversaries.

**Author Contributions:** Conceptualization, Y.Y. and K.W.; methodology, Y.Y.; software, Y.Y.; validation, K.Q. and W.W.; formal analysis, W.W.; investigation, K.Q.; resources, K.W.; data curation, Y.Y.; writing—original draft preparation, Y.Y.; writing—review and editing, W.W.; visualization, K.Q.; supervision, K.W.; project administration, K.W.; funding acquisition, K.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Scientific Research Startup Fund for Shenzhen High-Caliber Personnel of SZPT, No. 6022310051K, and the Industry-University-Research Innovation Fund for Chinese Universities (No. 2023IT068).

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Akyildiz, I.F.; Su, W.; Sankarasubramaniam, Y.; Cayirci, E. Wireless sensor networks: A survey. *Comput. Netw.* **2002**, *38*, 393–422. [\[CrossRef\]](#)
2. Ahmim, M.; Ouafi, N.; Ullah, I.; Ahmim, A.; Chefrour, D.; Almukhlifi, R. LSAP-IoHT: Lightweight Secure Authentication Protocol for the Internet of Healthcare Things. *Comput. Mater. Contin.* **2025**, *85*, 5093–5116. [\[CrossRef\]](#)
3. Huang, W. ECC-based three-factor authentication and key agreement scheme for wireless sensor networks. *Sci. Rep.* **2024**, *14*, 1787. [\[CrossRef\]](#)
4. Singh, M.; Mishra, D. Post-quantum secure authenticated key agreement protocol for wireless sensor networks. *Telecommun. Syst.* **2023**, *84*, 101–113. [\[CrossRef\]](#)
5. Roman, R.; Zhou, J.; Lopez, J. On the features and challenges of security and privacy in distributed internet of things. *Comput. Netw.* **2013**, *57*, 2266–2279. [\[CrossRef\]](#)
6. Zhou, Y.; Chen, L.; Zhao, X.; Yang, Z. An anonymous authentication scheme with controllable linkability for vehicle sensor networks. *Comput. Model. Eng. Sci.* **2020**, *125*, 1101–1118. [\[CrossRef\]](#)
7. Li, M.; Hu, S. A lightweight ECC-based authentication and key agreement protocol for IoT with dynamic authentication credentials. *Sensors* **2024**, *24*, 7967. [\[CrossRef\]](#)
8. Tyagi, G.; Kumar, R. An efficient user authentication and key agreement scheme for wireless sensor networks using physically unclonable function. *Int. J. Inf. Secur.* **2024**, *23*, 935–962. [\[CrossRef\]](#)

9. Li, Y.; Chen, Z.; Wang, H.; Sun, K.; Jajodia, S. Understanding account recovery in the wild and its security implications. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 620–634. [\[CrossRef\]](#)
10. Deng, M.; Ma, Q.; Song, Q.; Zhang, C.; Zuo, Z. Enhanced wireless sensor network authentication key agreement protocol. *Comput. Eng.* **2025**, *51*, 186–193. [\[CrossRef\]](#)
11. Feng, H.; Cai, B. A Provably Secure and Lightweight Two-Factor Authentication Protocol for Wireless Sensor Network. *Electronics* **2024**, *13*, 4289. [\[CrossRef\]](#)
12. Sharmila, A.; Rishiwal, V.; Kumar, P.; Yadav, M.; Yadav, P. Secure Hybrid Data Transmission Protocol for WSN with Key Management and Message Authentication. *SN Comput. Sci.* **2025**, *6*, 401. [\[CrossRef\]](#)
13. Dash, S.; Khan, A.U.; Kar, B.; Swain, S.K.; Kuswirdyo, P.; Tadele, S.B.; Wakgra, F.G. CRAMP: Clustering-based RANs association and MEC placement for delay-sensitive applications. *J. Netw. Comput. Appl.* **2024**, *227*, 103893. [\[CrossRef\]](#)
14. Yang, J.H. A multi-gateway authentication and key-agreement scheme on wireless sensor networks for IoT. *EURASIP J. Inf. Secur.* **2023**, *2023*, 2. [\[CrossRef\]](#)
15. Wang, L.; Han, C. Multi-factor authentication and key agreement scheme based on PUF and Chebyshev chaotic map for wireless sensor networks. *Sci. Rep.* **2025**, *16*, 3311. [\[CrossRef\]](#)
16. Hayouni, H. Adaptive post-quantum security framework for wireless sensor networks using lightweight cryptography and context-aware key management. *J. Supercomput.* **2025**, *81*, 1426. [\[CrossRef\]](#)
17. Paul, R.; Rai, S.; Banerjee, S.; Meher, P. A robust smart card based authentication and key agreement scheme for wsn using fuzzy Extractor. *Peer-to-Peer Netw. Appl.* **2024**, *17*, 432–450. [\[CrossRef\]](#)
18. Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Trans. Inf. Theory* **2003**, *29*, 198–208. [\[CrossRef\]](#)
19. Gassend, B.; Clarke, D.; Van Dijk, M.; Devadas, S. Silicon physical random functions. In Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington, DC, USA, 18–22 November 2002; pp. 148–160.
20. Burrows, M.; Abadi, M.; Needham, R. A logic of authentication. *ACM Trans. Comput. Syst. (TOCS)* **1990**, *8*, 18–36. [\[CrossRef\]](#)
21. Blanchet, B. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In Proceedings of the 14th IEEE Computer Security Foundations Workshop (CSFW-14), Washington, DC, USA, 11–13 June 2001; pp. 82–96.
22. AKA.pv. Available online: <https://github.com/yycitbT/ProVerif/blob/main/AKA.pv> (accessed on 26 December 2025).
23. Zhao, X.; Li, D.; Li, H. Practical three-factor authentication protocol based on elliptic curve cryptography for industrial internet of things. *Sensors* **2022**, *22*, 7510. [\[CrossRef\]](#)
24. Vangala, A.; Das, A.K.; Lee, J.H. Provably secure signature-based anonymous user authentication protocol in an Internet of Things-enabled intelligent precision agricultural environment. *Concurr. Comput. Pract. Exp.* **2023**, *35*, e6187. [\[CrossRef\]](#)
25. Kumar, C.M.; Dwivedi, S.K.; Brindha, M.; Al-Shehari, T.; Alfakih, T.; Alsalman, H.; Amin, R. REPACA: Robust ECC based privacy-controlled mutual authentication and session key sharing protocol in coalmines application with provable security. *Peer-to-Peer Netw. Appl.* **2024**, *17*, 4264–4285. [\[CrossRef\]](#)
26. Srinivas, J.; Das, A.K.; Wazid, M.; Vasilakos, A.V. Designing secure user authentication protocol for big data collection in IoT-based intelligent transportation system. *IEEE Internet Things J.* **2020**, *8*, 7727–7744. [\[CrossRef\]](#)

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.