symmetry

MDPI

Article

A Light and Anonymous Three-Factor Authentication Protocol for Wireless Sensor Networks

Lianghong Zhu [1,†], Huaikun Xiang [1,†] and Kai Zhang [2,*]

[1] School of Automotive and Transportation, Shenzhen Polytechnic, Shenzhen 518055, China; zhulianghong@szpt.edu.cn (L.Z.); xianghuaikun@szpt.edu.cn (H.X.)
[2] Department of Computer Science and Engineering, National Chung Hsing University, Taichung 402, Taiwan
* Correspondence: kai.zhang@smail.nchu.edu.tw
† These authors contributed equally to this work.

Abstract: Recently, wireless sensor networks (*WSNs*) have been widely used in a variety of fields, and make people's lives more convenient and efficient. However, *WSNs* are usually deployed in a harsh and insecure environment. Furthermore, sensors with limited hardware resources have a low capacity for data processing and communication. For these reasons, research on efficient and secure real-time authentication and key agreement protocols based on the characteristics of *WSNs* has gradually attracted the attention of academics. Although many schemes have been proposed, most of them cannot achieve all known security features with satisfactory performance, among which anonymity, N-Factor security, and forward secrecy are the most vulnerable. In order to solve these shortcomings, we propose a new lightweight and anonymous three-factor authentication scheme based on symmetric cryptographic primitives for *WSNs*. By using the automated security verification tool *ProVerif*, *BAN*-logic verification, and an informal security analysis, we prove that our proposed scheme is secure and realizes all known security features in *WSNs*. Moreover, we show that our proposed scheme is practical and efficient through the comparison of security features and performance.

Keywords: authentication and key agreement; symmetric cryptographic primitives; three-factor; security analysis; wireless sensor networks

Citation: Zhu, L.; Xiang, H.; Zhang, K. A Light and Anonymous Three-Factor Authentication Protocol for Wireless Sensor Networks. *Symmetry* **2022**, *14*, 46. https://doi.org/10.3390/sym14010046

Academic Editors: Alexander Zaslavski, Yi Fang, Long Shi and Pingping Chen

Received: 19 October 2021
Accepted: 27 December 2021
Published: 30 December 2021

1. Introduction

A wireless sensor network (*WSN*) is a distributed and self-organizing sensor network, which is composed of a large number of sensor nodes that can perceive and understand the external world. In *WSNs*, sensor nodes cooperate to sense, collect, and process information in the network coverage area and send it to the gateway node. In recent years, *WSNs* have been widely used in various practical applications in industrial and agricultural fields [1–4], such as temperature monitoring in agriculture, power consumption monitoring in smart grids, and human health monitoring in medical care. In these application environments, many scattered users, various randomly distributed sensor nodes, and one or more relatively powerful gateway nodes form a powerful network system. For example, in the field of health care, the sensors deployed on the patient's body can monitor and obtain the patient's body data, and the medical staff can directly and remotely obtain the patient's current body temperature, blood pressure, pulse times, and other information in real-time through the wireless sensor network, so as to improve the health status of healthy patients. Figure 1 shows the network architecture of *WSNs*.

However, when sensor nodes are active in an unattended or hostile wireless network environment, attackers can easily intercept, delete, and modify transmission messages and launch various attacks [5]. Therefore, the network security and privacy of these sensors become very critical. In order to ensure that only authorized legitimate users can access sensors and protect the communication security of real-time sensing data, it is extremely

necessary for users and sensors to authenticate each other directly. Moreover, they also need to be able to establish a session key to ensure the security of future communication. Authentication and key agreement protocols are effective ways to achieve these goals. However, due to the limited resources of sensor nodes, an authentication protocol based on complex asymmetric cryptographic primitives is difficult to apply to wireless sensor networks. Therefore, the balance between security and performance is highly significant for the design of identity authentication protocols in the wireless sensor network environment.

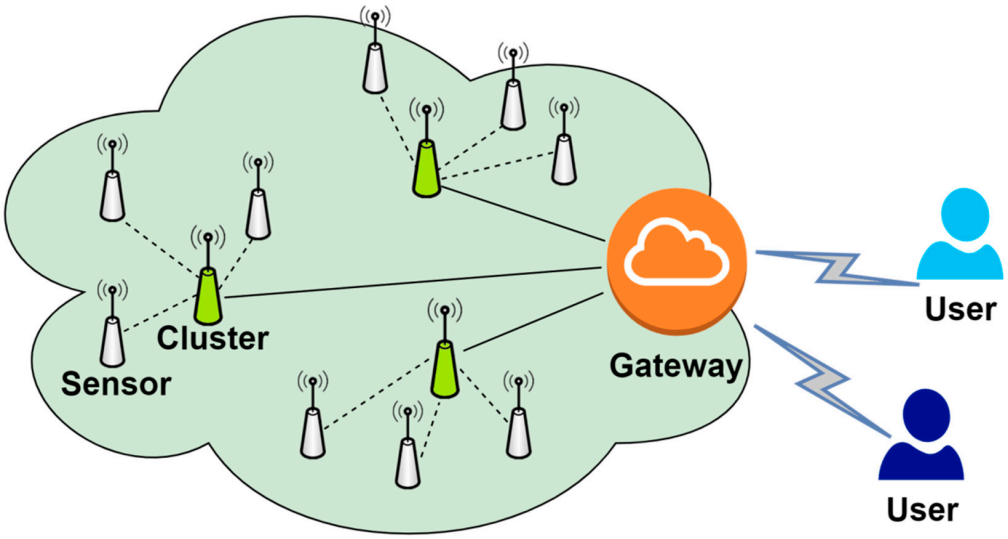

Figure 1. Network model of wireless sensor networks.

Lamport proposed the first password-based authentication and key agreement protocol in 1980 [6], and, since then, research on authentication protocols has been a hot topic. In recent years, research on authentication and key agreement protocols in *WSNs* has been conducted [7–18]. In 2006, Wong et al. [19] proposed a lightweight password-based authentication scheme for *WSNs*. However, Das et al. point out that Wong et al.'s scheme cannot resist replay attacks and stolen-verifier attacks [20]. Furthermore, Das et al. put forward an improved scheme. Unfortunately, there are also many security flaws in Das et al.'s scheme [21,22]. Based on Farash et al.'s scheme [23], Amin et al. provided an anonymity three-factor authentication scheme [24] for *WSNs* in 2016. However, Jiang et al. found that the scheme could not resist smart card stolen attacks and known session-specific temporary information attacks [25]. Although many schemes have been proposed, most of them cannot achieve all known security features with satisfactory performance.

In 2019, Shin et al. proposed a lightweight three-factor authentication and key agreement protocol for *WSNs* [26] and claimed that the protocol can achieve all known security features with satisfactory performance. This article analyzes Shin et al.'s scheme. It is found that their scheme is vulnerable to de-synchronization attacks and cannot achieve forward secrecy and three-factor security.

Our crucial contributions are as follows:

1. We review and analyze Shin et al.'s three-factor authentication scheme for *WSNs*. Further, we show that their scheme is vulnerable to de-synchronization attacks and cannot achieve forward secrecy and three-factor security.
2. We present a new, lightweight anonymous three-factor authentication with perfect forward secrecy in *WSNs*. The operation of the scheme is based on a symmetric cryptosystem, so the computational overhead of the scheme is lightweight and the scheme is suitable for *WSNs*. The new scheme can achieve all known strong security functions with satisfactory performance, including anonymity, perfect forward secrecy, n-factor security, and so on.

3. By using the automated security verification tool *ProVerif* and *BAN*-logic, we prove that our proposed scheme is secure and realizes the mutual authentication of communication participants in *WSNs*.
4. Through the comparison of security features and performance, it can be found that our proposed scheme is practical.

The rest of this paper is organized as follows. We introduce the relevant preliminaries in Section 2 and review the scheme of Shin et al. in Section 3. In Section 4, the scheme of Shin et al. is subjected to cryptanalysis and the attack method is given. The new scheme is proposed in Section 5, and the security analysis and performance analysis of the new scheme are carried out in Sections 6 and 7, respectively. Section 8 summarizes the paper.

2. Preliminaries

2.1. Fuzzy Extractor

At different times, there may be subtle differences in the biometrics extracted by the same user. The fuzzy extractor can eliminate these subtle differences. In other words, the fuzzy extractor can produce the same output even if the inputs are slightly different. Fuzzy logic is widely used in supply chains and healthcare logistics [27–29]. The fuzzy extractor consists of two parts:

1. $GEN(Bio_i) = (b_i, pair_i)$, with Bio_i. as input, the probability generation mechanism GEN outputs a random string b_i and a random helper string $pair_i$.
2. $REP(Bio_i', pair_i) = b_i$, with Bio_i' and $pair_i$ as inputs, the deterministic mechanism REP can regenerate b_i, where $dis(Bio_i', Bio_i) \leq \Delta t$.

2.2. Adversary Model

In this paper, we adopt the most rigorous (but practical) adversary model proposed by Wang et al. [30] and Huang et al. [31]. Table 1 shows the capabilities of the adversary, A.

Table 1. The capabilities of the adversary.

Symbol	Description
C1	A can enumerate every possibility of user identity and password.
C2	A can eavesdrop, intercept, insert, delete, or block messages transmitted in the public channel.
C3	For a three-factor protocol (password, smart card, and biometric), A can capture two of the authentication factors simultaneously.
C4	Expired session keys can be captured.
C5	A can obtain the long-term private keys of users, *GWNs*, or sensors (only when evaluating forward secrecy).

2.3. Notations

The notations used thereinafter are listed in Table 2.

Table 2. Notations.

Symbol	Description
GWN	Gateway Node
U_i	User
ID_i	Identification of U_i
PW_i	Password of U_i
SC_i	Smart card of U_i
Bio_i	Biometric of U_i
b_i	Random string generated by a fuzzy extractor
$pair_i$	Random helper string generated by a fuzzy extractor
PID_i	Pseudo identification of U_i

Table 2. *Cont.*

Symbol	Description
S_j	Sensor Node
SID_j	Identification of S_j
SK_{ij}	Session Key of U_i and S_j
T_1, T_2, T_3, T_4	Timestamp
$\bigoplus$	*XOR* Operation
$H(*)/h(*)$	Hash Function
$\|\|$	Concatenation operation

3. Revisiting of Shin et al.'s Scheme

In 2019, Sooyeon Shin et al. proposed a lightweight anonymous three-factor authentication protocol for micro-sensors in wireless sensor networks [26]. Taking their protocol as an example, we analyze and point out the security defects of such authentication protocols.

Shin et al.'s protocol consists of four phases: the initialization phase, registration phase, authentication phase, and password update phase. The system completes the selection of parameters and encryption algorithm in the initialization phase. The registration phase realizes user registration and the distribution of smart cards. The authentication phase completes the mutual authentication and session key agreement between the user and the sensor. It should be noted that the communication channel in the authentication phase is public and insecure.

The specific process of the agreement is as follows:

3.1. Initialization Phase

Step 1: The *GWN* selects K_U, K_S as master secrets and stores them safely.

Step 2: For sensor S_j, the *GWN* chooses SID_j as the identity of S_j and calculates $X_{S_j} = h(SID_j||K_S)$.

Step 3: The sensor S_j stores X_{S_j} secretly.

3.2. User Registration Phase

Step 1: The user U_i. selects his identity ID_i and password PW_i and imprints Bio_i. Then, U_i chooses a random number u_i and calculates $GEN(Bio_i) = (b_i, pair_i)$, $HPW_i = h(PW_i||b_i)$, and $TID_i = h(ID_i||u_i)$. Further, U_i sends the registration request $\{TID_i, HPW_i\}$ towards the *GWN* via a private secure channel.

Step 2: The *GWN* receives $\{TID_i, HPW_i\}$ and freely chooses PID_i^1. as a pseudonym. Then, the *GWN* calculates $HID_i = h(TID_i||K_U)$, $A_i = h(HPW_i||TID_i) \bigoplus HID_i$, $B_i = h(HPW_i||HID_i)$, and $C_i^1 = h(TID_i||HID_i) \bigoplus PID_i^1$. Then, the *GWN* writes $\{A_i, B_i, C_i^1\}$ into SC_i. and stores (PID_i^1, TID_i). Finally, the *GWN* transmits SC_i towards U_i via a private, secure channel.

Step 3: U_i receives SC_i and calculates $D_i = u_i \bigoplus h(ID_i||b_i)$. Finally, U_i write $\{D_i, pair_i\}$ into SC_i.

The process of the User registration phase is shown in Figure 2.

3.3. Authentication Phase

Step 1: U_i inserts SC_i, inputs ID_i and PW_i, and imprints Bio_i. SC_i calculates $b_i = REP(Bio_i, pair_i)$, $u_i = D_i \bigoplus h(ID_i||b_i)$, $TID_i = h(ID_i||u_i)$, $HPW_i^* = h(PW_i||b_i)$, $HID_i^* = A_i \bigoplus h(HPW_i^*||TID_i)$, and $B_i^* = h(HPW_i^*||HID_i^*)$ and verifies the equality check for $B_i ? = B_i^*$. If it does not hold true, SC_i rejects the login request. Otherwise, SC_i generates a random number r_i and the current timestamp T_1, and calculates $PID_i^1 = C_i^1 \bigoplus h(TID_i||HID_i^*)$, $R_i = h(TID_i||PID_i^1||r_i)$, $M_i = r_i \bigoplus h(TID_i||HID_i^*||T_1)$, and $M_{UG} = h(TID_i||HID_i^*||PID_i^1||R_i||T_1)$. Finally, SC_i transmits the login request $\{PID_i^1, M_i, M_{UG}, T_1\}$ towards the *GWN*.

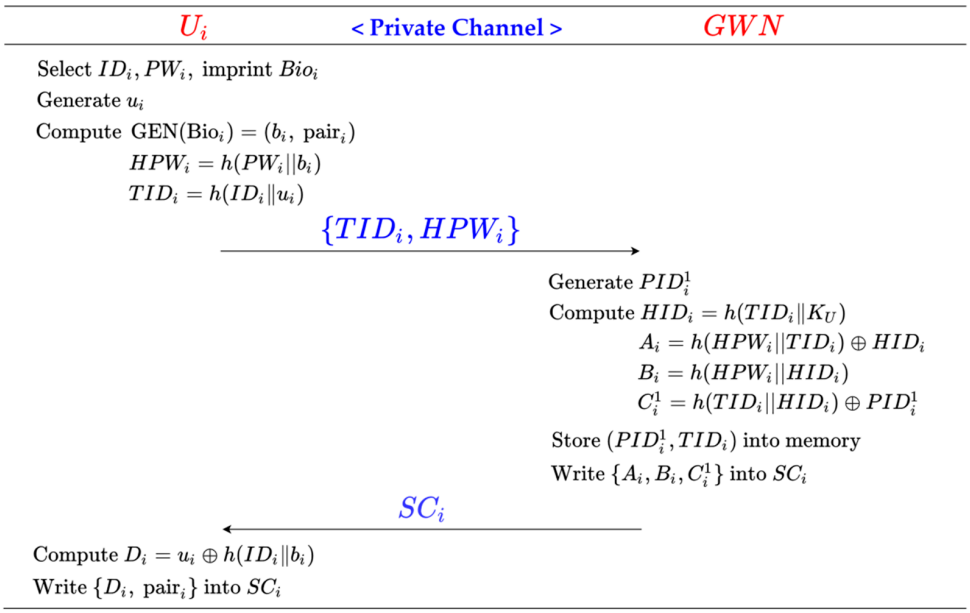

Figure 2. User registration phase of Shin et al.'s scheme.

Step 2: The *GWN* receives $\{PID_i^1, M_i, M_{UG}, T_1\}$ and checks the validity of T_1. Then, the *GWN* searches (PID_i^1, TID_i) in memory using PID_i^1 and calculates $HID_i^* = h(TID_i||K_U)$, $r_i^* = M_i \bigoplus h(TID_i||HID_i^*||T_1)$, $R_i^* = h(TID_i||PID_i^1||r_i^*)$, and $M_{UG}^* = h(TID_i||HID_i^*||PID_i^1||R_i^*||T_1)$. Further, the *GWN* checks the equality of $M_{UG}^* \stackrel{?}{=} M_{UG}$. If it does not hold true, the *GWN* rejects the login request. Otherwise, the *GWN* selects SID_j, generates the current timestamp T_2, and calculates $X_{S_j} = h(SID_j||K_S)$, $M_G = R_i^* \bigoplus h(X_{S_j}||T_2)$, and $M_{\mathrm{GS}} = h(PID_i^1||SID_j||X_{S_j}||R_i^*||T_2)$. Finally, the *GWN* transmits $\{PID_i^1, M_G, M_{\mathrm{GS}}, T_2\}$ towards S_j.

Step 3: Upon the reception of $\{PID_i^1, M_G, M_{\mathrm{GS}}, T_2\}$ from the *GWN*, S_j checks whether T_2 is a valid timestamp. Then, S_j calculates $R_i^* = M_G \bigoplus h(X_{S_j}||T_2)$ and $M_{\mathrm{GS}}^* = h(PID_i^1||SID_j||X_{S_j}||R_i^*||T_2)$ and verifies the equality check $M_{\mathrm{GS}}^* \stackrel{?}{=} M_{\mathrm{GS}}$. If the verification fails, S_j aborts the session. Otherwise, S_j generates a random number r_j and the current timestamp T_3, and calculates $R_j = h(SID_j||r_j)$, $M_j = r_j \bigoplus h(X_{S_j}||T_3)$, $SK_{ij} = h(R_i^*||R_j)$, $M_{\mathrm{SG}} = h(PID_i^1||SID_j||X_{S_j}||R_j||SK_{ij}||T_3)$. Finally, S_j sends $\{M_j, M_{\mathrm{SG}}, T_3\}$ back to the *GWN*.

Step 4: The *GWN* receives $\{M_j, M_{\mathrm{SG}}, T_3\}$ and checks the validity of T_3. Then, the *GWN* calculates $r_j^* = M_j \bigoplus h(X_{S_j}||T_3)$, $R_j^* = h(SID_j||r_j^*)$,$SK_{ij}^* = h(R_i^*||R_j^*)$, and $M_{\mathrm{SG}}^* = h(PID_i^1||SID_j||X_{S_j}||R_j^*||SK_{ij}^*||T_3)$ and verifies the equality check $M_{\mathrm{SG}}^* \stackrel{?}{=} M_{\mathrm{SG}}$. If the verification fails, the *GWN* aborts the session. Otherwise, the *GWN* generates the current timestamp T_4 and a new pseudonym PID_i^2, and calculates $C_i^2 = h(TID_i||HID_i^*) \bigoplus PID_i^2$, $p_i^2 = C_i^2 \bigoplus h(HID_i^*||T_4)$, $M_G' = R_j^* \bigoplus h(PID_i^1||HID_i^*)$, and $M_{\mathrm{GU}} = h(PID_i^1||HID_i^*||C_i^2||R_j^*||SK_{ij}^*||T_4)$. Finally, the *GWN* sends $\{p_i^2, M_G', M_{\mathrm{GU}}, T_4\}$ back to U_i. and replaces PID_i^1 with PID_i^2 in memory.

Step 5: U_i receives $\{p_i^2, M_G', M_{\mathrm{GU}}, T_4\}$ and checks whether T_4 is a valid timestamp. Then, U_i calculates $R_j^* = M_G' \bigoplus h(PID_i^1||HID_i^*)$, $SK_{ij}^* = h(R_i||R_j^*)$, $C_i^2 = p_i^2 \bigoplus h(HID_i^*||T_4)$, and $M_{\mathrm{GU}}^* = h(PID_i^1||HID_i^*||C_i^2||R_j^*||SK_{ij}^*||T_4)$ and checks the equality of $M_{\mathrm{GU}}^* \stackrel{?}{=} M_{\mathrm{GU}}$. If it holds true, U_i replaces C_i^1 with C_i^2 in SC_i.

The process of the authentication phase is shown in Figure 3.

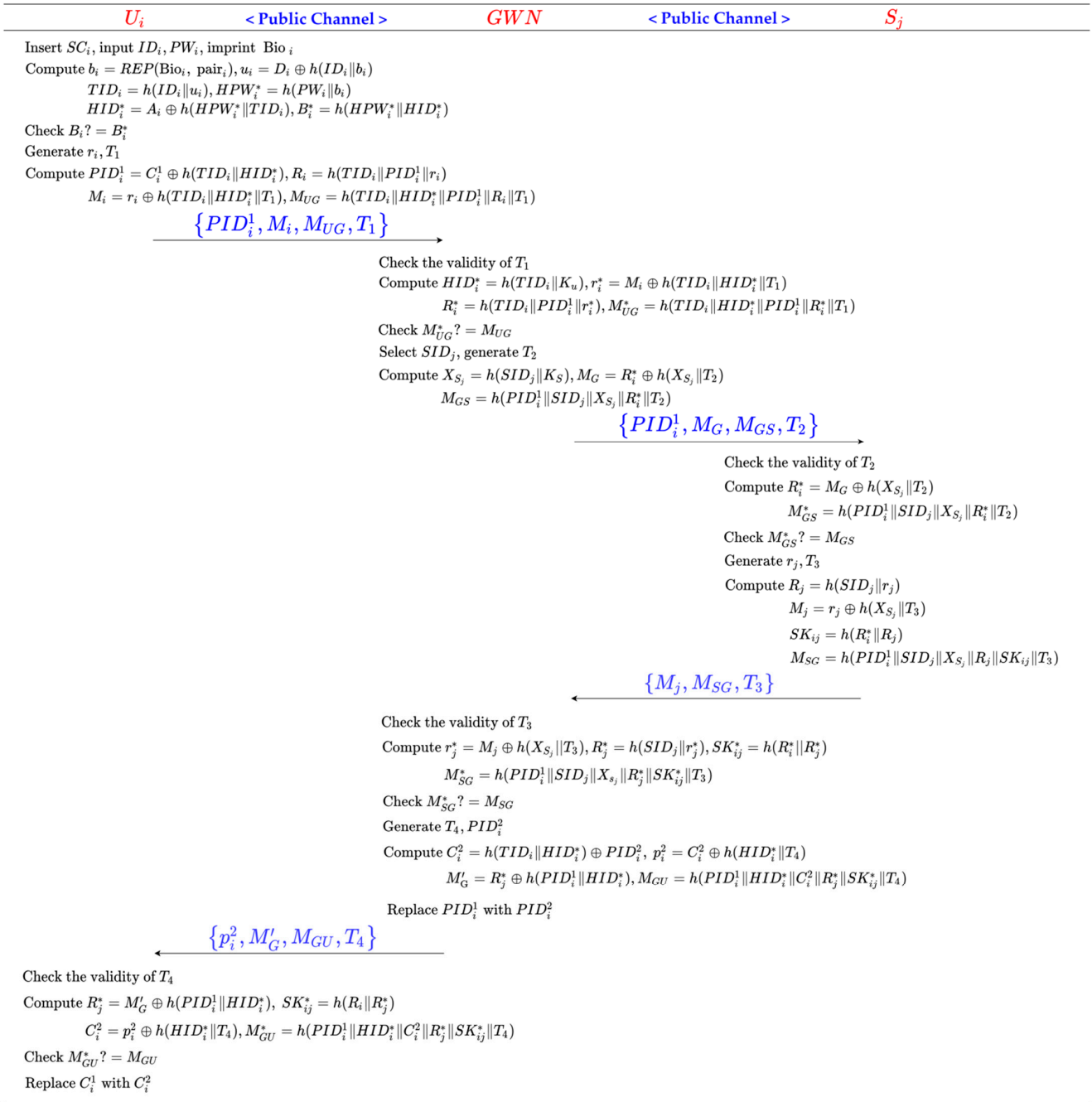

Figure 3. Authentication phase of Shin et al.'s scheme.

3.4. Password Update Phase

Step 1: U_i inserts SC_i to the reader, inputs ID_i and PW_i, and imprints Bio_i.

Step 2: SC_i calculates $b_i = REP(Bio_i, pair_i)$, $u_i = D_i \bigoplus h(ID_i||b_i)$, $TID_i = h(ID_i||u_i)$, $HPW_i^* = h(PW_i||b_i)$, $HID_i^* = A_i \bigoplus h(HPW_i^*||TID_i)$, and $B_i^* = h(HPW_i^*||HID_i^*)$ and verifies the equality check for $B_i ? = B_i^*$. If it does not hold true, SC_i rejects the request. Otherwise, U_i inputs a new password PW_i^{new}.

Step 3: SC_i calculates $HPW_i^{new} = h(PW_i^{new}||b_i)$, $A_i^{new} = h(HPW_i^{new}||TID_i) \bigoplus HID_i^*$, and $B_i^{new} = h(HPW_i^{new}||HID_i^*)$. At last, SC_i replaces A_i and B_i with A_i^{new} and B_i^{new}, respectively.

4. Cryptanalysis of Shin et al.'s Scheme

We show that Shin et al.'s scheme is vulnerable to de-synchronization attacks and can not achieve forward secrecy and three-factor security in this section.

4.1. De-Synchronization Attack

Suppose that an adversary blocks $\{p_i^2,\ M'_G,\ M_{\mathrm{GU}},\ T_4\}$, which is sent from the *GWN* to U_i. On the side of the *GWN*, the pseudonym of U_i is PID_i^2 at this point. However, U_i is unable to obtain C_i^2 without $\{p_i^2,\ M'_G,\ M_{\mathrm{GU}},\ T_4\}$. Thus, the pseudonyms on the side of U_i and the *GWN* become out of synchronization. When U_i wants to access a sensor node through the *GWN* in the next session, the *GWN* will reject U_i's login request. Therefore, Shin et al.'s scheme is vulnerable to de-synchronization attacks.

4.2. Forward Secrecy

Suppose that an adversary occasionally obtains X_{S_j}, which is the long-term private key of S_j. Furthermore, the adversary intercepted $\{PID_i^1,\ M_G,\ M_{\mathrm{GS}},\ T_2\}$ and $\{M_j,\ M_{\mathrm{SG}},\ T_3\}$ in the previous session of U_i and S_j. The adversary could obtain the previous session key of U_i and S_j via following steps.

Step 1: The adversary calculates $R_i^* = M_G \bigoplus h(X_{S_j}||T_2)$, $r_j^* = M_j \bigoplus h(X_{S_j}||T_3)$, and $R_j^* = h(SID_j||r_j^*)$.

Step 2: The adversary obtains the previous session key of U_i and S_j, $SK_{ij} = h(R_i^*||R_j^*)$.

Therefore, Shin et al.'s scheme can not achieve forward secrecy.

4.3. Three-Factor Security

For a three-factor authentication scheme, when two of the authentication factors are captured by an adversary, it is necessary to ensure that the remaining authentication factor is still secure. Suppose that an adversary captures U_i's smart card SC_i and biometric Bio_i simultaneously. The adversary is able to obtain the password of U_i via the following steps.

Step 1: The adversary extracts $\{A_i,\ B_i,\ C_i^1,\ D_i,\ pair_i\}$ from SC_i using side-channel technology and calculates $b_i = REP(Bio_i,\ pair_i)$.

Step 2: The adversary guesses a candidate identity ID_i^* and a candidate password PW_i^* from D_{id} and D_{pw}, where D_{id} and D_{pw} are user identity space and password space, respectively.

Step 3: The adversary calculates $u_i^* = D_i \bigoplus h(ID_i^*||b_i)$, $TID_i^* = h(ID_i^*||u_i^*)$, $HPW_i^* = h(PW_i^*||b_i)$, $HID_i^* = A_i \bigoplus h(HPW_i^*||TID_i^*)$, and $B_i^* = h(HPW_i^*||HID_i^*)$.

Step 4: The adversary checks whether $B_i \stackrel{?}{=} B_i^*$ holds. If not, the adversary repeats Steps 2–4 until he acquires the true password. Otherwise, the adversary succeeds in obtaining the true password of U_i.

The computational overhead of this attack is $(5T_h + 2T_{xor}) * |D_{id}| * |D_{pw}|$, where T_h is the running time of the one-way hash function, T_{xor} is the running time of the *XOR* operation, and D_{id} and D_{pw} are the spaces of user identity and password, respectively. According to the literature [20], we know that $|D_{id}| \leq |D_{pw}| \leq 10^6$. According to the experimental data from the literature [32], $T_h \approx 0.591\mu s$, $T_{xor} \approx 0.006\ \mu s$. The adversary can break the password of U_i in 35 days. If you use a high-performance cloud computing platform, the password will be cracked within a few hours.

5. The Proposed Scheme

The proposed protocol includes the following phases: initialization phase, user registration phase, sensor node registration phase, authentication phase, password, and biometric update phase.

The detailed description of the agreement is as follows:

5.1. Initialization Phase

The gateway node *GWN* creates two information tables in its memory (*UserInfoTable* and *SensorInfoTable*), which is used to store relevant information of users and sensors.

Then, the *GWN* freely chooses two master keys, K_u and K_s, and two secure hash functions, $h : \{0,1\}^* \bigoplus \{0,\ 1\}^{128}$ and $H : \{0,\ 1\}^* \bigoplus \{0,\ 1\}^{256}$.

5.2. Sensor Registration Phase

The sensor registration phase is completed by the gateway node *GWN*. The *GWN* selects a unique identity SID_j for each sensor node and calculates $X_{S_j} = h(SID_j||K_s)$. Furthermore, *GWN* selects two random integers, n_j and c, defines and sets $N_j = NG_j = c$. Then, the *GWN* inserts the $\{SID_j,\ NG_j,\ X_{S_j},\ n_j\}$ into *SensorInfoTable* in its memory. Before S_j is deployed, the *GWN* stores $\{SID_j,\ N_j,\ X_{S_j},\ n_j\}$ into S_j.

5.3. User Registration Phase

Step 1: U_i chooses ID_i and PW_i freely, imprints Bio_i. Then U_i calculates $GEN(Bio_i) = (b_i,\ pair_i)$, $HPW_i = h(PW_i||b_i)$, and $TID_i = h(ID_i)$. Finally, U_i sends the registration request $\{TID_i,\ HPW_i\}$ towards the *GWN* via a private secure channel.

Step 2: The *GWN* receives $\{TID_i,\ HPW_i\}$ and checks if *UserInfoTable*() contains the element $(TID_i,\ *,\ *,\ *,\ *)$. If yes, the *GWN* rejects the registration request of U_i. Otherwise, the *GWN* chooses $a,\ b$ randomly, and sets $NC_i = a, PID_i = PID_i^{new} = b, PID_i^{old} = Null$. Then, the *GWN* calculates $HID_i = h(TID_i||K_u)$, $A_i = h(HPW_i||TID_i) \bigoplus HID_i$, $B_i = h(HPW_i||HID_i)$ *mod* n, and $C_i = h(TID_i||HID_i) \bigoplus PID_i$, where $2^4 \leq n \leq 2^8$ is an integer to determine the size of $(ID,\ PW)$, and inserts the element $(PID_i^{new},\ PID_i^{old},\ TID_i,\ NC_i)$ into table *UserInfoTable*. Further, the *GWN* writes $\{A_i,\ b_i,\ C_i,\ NC_i,\ h(\cdot),\ H(\cdot)\}$ into SC_i, and transmits SC_i towards U_i via a private secure channel.

Step 3: U_i receives SC_i, and defines and sets $flag = 0$. Finally, U_i writes $\{pair_i,\ flag,\ GEN(\cdot),\ REP(\cdot)\}$ into SC_i.

The process of the user registration phase is shown in Figure 4.

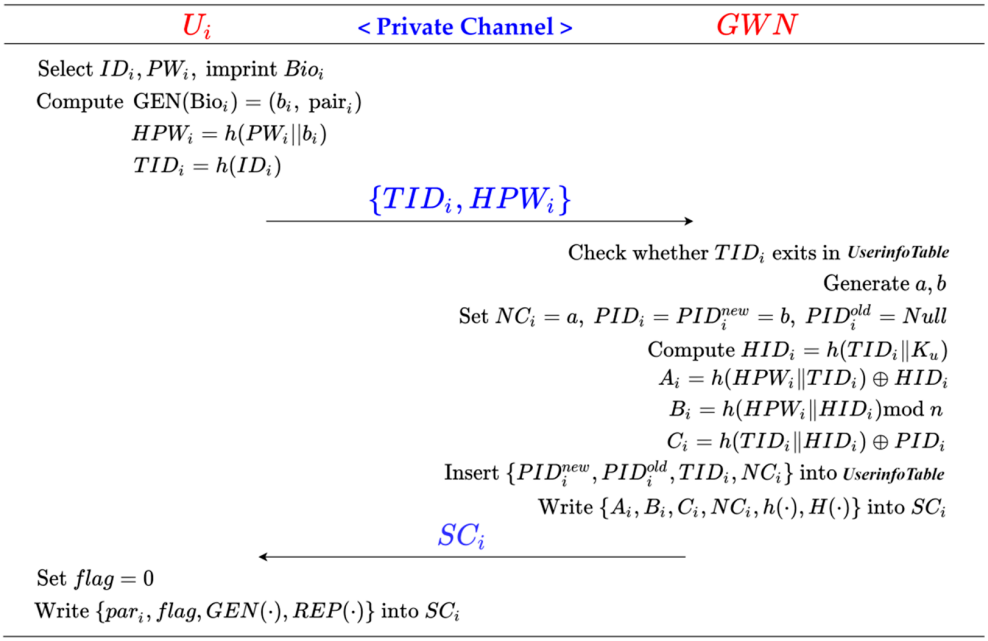

Figure 4. User registration phase of the proposed scheme.

5.4. Authentication Phase

Step 1: U_i inserts SC_i, inputs ID_i and PW_i, and imprints Bio_i. SC_i calculates $b_i = REP(Bio_i,\ pair_i)$, $TID_i = h(ID_i)$, $HPW_i^* = h(PW_i||b_i)$, $HID_i^* = A_i \bigoplus h(HPW_i^*||TID_i)$, and $B_i^* = h(HPW_i^*||HID_i^*)$ *mod* n and verifies the equality check for $B_i^*\ ? = B_i$. If it does not hold true, SC_i rejects the login request. Otherwise, SC_i checks if $flag\ ? = 0$ holds.

If yes, SC_i updates $NC_i = h(NC_i)$, $flag = 1$. Then, SC_i generates the current timestamp T_1 and a random number r_i, chooses SID_j which he wants to access and calculates $PID_i = C_i \oplus h(TID_i||HID_i^*)$, $R_i = h(TID_i||PID_i||NC_i||r_i)$, $M_1 = (r_i||SID_j) \oplus H(TID_i||HID_i^*||NC_i||T_1)$, and $M_{UG} = h(TID_i||HID_i^*||PID_i||R_i||T_1)$. Finally, SC_i transmits the login request $\{PID_i, M_1, M_{UG}, T_1\}$ towards the *GWN*.

Step 2: The *GWN* receives $\{PID_i, M_1, M_{UG}, T_1\}$ and checks the validity of T_1. Then, the *GWN* searches $(PID_i^{new}, PID_i^{old}, TID_i, NC_i)$ in *UserInfoTable* using PID_i and operates as below.

Case 1: If there exists an element $(PID_i^{new}, PID_i^{old}, TID_i, NC_i)$ of *UserInfoTable* which satisfies $PID_i = PID_i^{new}$, then the *GWN* calculates $NC_i' = h(NC_i)$, $HID_i^* = h(TID_i||K_u)$, $K_i = H(TID_i||HID_i^*||NC_i'||T_1)$, $(r_i^*||SID_j) = K_i \oplus M_1$, $R_i^* = h(TID_i||PID_i^{new}||NC_i'||r_i^*)$, and $M_{UG}^* = h(TID_i||HID_i^*||PID_i^{new}||R_i^*||T_1)$. The *GWN* verifies the equality check $M_{UG}^* \stackrel{?}{=} M_{UG}$. If the verification fails, the *GWN* rejects the login request. Otherwise, the *GWN* chooses a new $tPID_i^{new}$ randomly, and sets $PID_i^{old} = PID_i^{new}$, $PID_i^{new} = tPID_i^{new}$, $NC_i = NC_i'$.

Case 2: If there exists an element $(PID_i^{new}, PID_i^{old}, TID_i, NC_i)$ of *UserInfoTable* which satisfies $PID_i = PID_i^{old}$, then the *GWN* calculates $HID_i^* = h(TID_i||K_u)$, $K_i = H(TID_i||HID_i^*||NC_i||T_1)$, $(r_i^*||SID_j) = K_i \oplus M_1$, $R_i^* = h(TID_i||PID_i^{old}||NC_i||r_i^*)$, and $M_{UG}^* = h(TID_i||HID_i^*||PID_i^{old}||R_i^*||T_1)$. The *GWN* verifies the equality check $M_{UG}^* \stackrel{?}{=} M_{UG}$. If the verification fails, the *GWN* rejects the login request. Otherwise, the *GWN* chooses a new $tPID_i^{new}$ randomly, and sets $PID_i^{new} = tPID_i^{new}$.

Case 3: If the above two cases do not exist, the *GWN* rejects the login request.

Further, the *GWN* generates the current timestamp T_2, searches $\{SID_j, NG_j, X_{S_j}, n_j\}$ in *SensorInfoTable* using SID_j, and updates $NG_j = NG_j + n_j$, $X_{S_j} = h(SID_j||X_{S_j})$. Then, the *GWN* calculates $M_2 = (R_i^*||PID_i^{old}) \oplus H(X_{S_j}||T_2)$ and $M_{\mathrm{GS}} = h(PID_i^{old}||SID_j||X_{S_j}||R_i^*||T_2)$. Finally, the *GWN* transmits $\{M_2, M_{\mathrm{GS}}, NG_j, T_2\}$ towards S_j.

Step 3: Upon the reception of $\{M_2, M_{\mathrm{GS}}, NG_j, T_2\}$ from the *GWN*, S_j checks whether T_2 is a valid timestamp. Then, S_j calculates $N' = NG_j - NS_j/n_j$ and checks if $1 \leq N' \leq N$ holds, where N is the initial threshold for preserving the computing resources of sensors. If it holds true, S_j sets $X_{S_j}' = X_{S_j}$ and calculates N' times $X_{S_j}' = h(X_{S_j}'||SID_j)$. Further, S_j calculates $(R_i^*||PID_i^{old}) = M_2 \oplus H(X_{S_j}'||T_2)$ and $M_{\mathrm{GS}}^* = h(PID_i^{old}||SID_j||X_{S_j}'||R_i^*||T_2)$ and verifies the equality check $M_{\mathrm{GS}}^* \stackrel{?}{=} M_{\mathrm{GS}}$. If the verification fails, S_j aborts the session. Otherwise, S_j generates the current timestamp T_3 and a random number r_j and calculates $R_j = h(SID_j||r_j)$, $M_3 = (R_j||PID_i^{old}) \oplus H(X_{S_j}'||T_3)$, $SK_{ji} = h(R_i^*||R_j)$, and $M_{\mathrm{SG}} = h(PID_i^{old}||SID_j||X_{S_j}'||R_i^*||R_j||T_3)$. S_j updates $X_{S_j} = X_{S_j}'$, $N_j = NG_j$. Finally, S_j sends $\{SID_j, M_3, M_{\mathrm{SG}}, T_3\}$ back to the *GWN*.

Step 4: The *GWN* receives $\{SID_j, M_3, M_{\mathrm{SG}}, T_3\}$ and checks the validity of T_3. Then, the *GWN* searches $\{SID_j, NG_j, X_{S_j}, n_j\}$ in *SensorInfoTable* using SID_j and calculates $(R_j||PID_i^{old}) = M_3 \oplus H(X_{S_j}||T_3)$, and $M_{\mathrm{SG}}^* = h(PID_i^{old}||SID_j||X_{S_j}||R_i^*||R_j^*||T_3)$. The *GWN* verifies the equality check $M_{\mathrm{SG}}^* \stackrel{?}{=} M_{\mathrm{SG}}$. If the verification fails, the *GWN* aborts the session. Otherwise, the *GWN* generates the current timestamp T_4 and searches $(PID_i^{new}, PID_i^{old}, TID_i, NC_i)$ in *UserInfoTable* using PID_i^{old}. Further, the *GWN* calculates $C_i^{new} = h(TID_i||HID_i^*) \oplus PID_i^{new}$, $p_i^{new} = C_i^{new} \oplus HID_i^* \oplus T_4$, $M_4 = R_j^* \oplus K_i$, and $M_{\mathrm{GU}} = h(PID_i^{old}||HID_i^*||C_i^{new}||R_i^*||R_j^*||T_4)$. Finally, the *GWN* sends $\{p_i^{new}, M_4, M_{\mathrm{GU}}, T_4\}$ to U_i.

Step 5: Upon the reception of $\{p_i^{new}, M_4, M_{\mathrm{GU}}, T_4\}$ from the *GWN*, U_i checks whether T_4 is a valid timestamp. Then, U_i calculates $R_j^* = M_4 \oplus H(TID_i||HID_i^*||NC_i||T_1)$, $SK_{ij} = h(R_i||R_j^*)$, $C_i^{new} = p_i^{new} \oplus HID_i^* \oplus T_4$, and $M_{\mathrm{GU}}^* = h(PID_i||HID_i^*||C_i^{new}||R_i||R_j^*||T_4)$. Then, U_i verifies the equality check $M_{\mathrm{GU}}^* \stackrel{?}{=} M_{\mathrm{GU}}$. If the verification fails, U_i aborts the session. Otherwise, U_i updates $C_i = C_i^{new}$, $flag = 0$.

The process of the authentication phase is shown in Figure 5.

U_i	< Public Channel >	GWN	< Public Channel >	S_j
Insert SC_i, input ID_i, PW_i, imprint Bio_i				
Compute $b_i = REP(Bio_i, \text{pair}_i), TID_i = h(ID_i)$				
$HPW_i^* = h(PW_i \Vert b_i), HID_i^* = A_i \oplus h(HPW_i^* \Vert TID_i)$				
$B_i^* = h(HPW_i^* \Vert HID_i^*) \bmod n$				
Check $B_i ? = B_i^*$				
If $flag = 0$, update $NC_i = h(NC_i)$, $flag = 1$				
Generate r_i, T_1, select SID_j				
Compute $PID_i = C_i \oplus h(TID_i \Vert HID_i^*), R_i = h(TID_i \Vert PID_i \Vert NC_i \Vert r_i)$				
$R_i = h(TID_i \Vert PID_i \Vert NC_i \Vert r_i), M_1 = (r_i \Vert SID_j) \oplus H(TID_i \Vert HID_i^* \Vert NC_i \Vert T_1)$				
$M_{UG} = h(TID_i \Vert HID_i^* \Vert PID_i \Vert R_i \Vert T_1)$				
	$\{PID_i, M_1, M_{UG}, T_1\} \rightarrow$			
		Check the validity of T_1		
		If $PID_i = PID_i^{new}$; Compute $NC_i' = h(NC_i)$, $HID_i^* = h(TID_i \Vert K_u)$		
		$K_i = H(TID_i \Vert HID_i^* \Vert NC_i' \Vert T_1), (r_i^* \Vert SID_j) = M_1 \oplus K_i$		
		$R_i^* = h(TID_i \Vert PID_i^{new} \Vert NC_i' \Vert r_i^*)$		
		$M_{UG}^* = h(TID_i \Vert HID_i^* \Vert PID_i^{new} \Vert R_i^* \Vert T_1)$		
		Check $M_{UG}^* ? = M_{UG}$		
		Generate $tPID_i^{new}$		
		Set $PID_i^{old} = PID_i^{new}$, $PID_i^{new} = tPID_i^{new}, NC_i = NC_i'$		
		If $PID_i = PID_i^{old}$; Compute $HID_i^* = h(TID_i \Vert K_u)$		
		$K_i = H(TID_i \Vert HID_i^* \Vert NC_i \Vert T_1), (r_i^* \Vert SID_j) = M_1 \oplus K_i$		
		$R_i^* = h(TID_i \Vert PID_i^{old} \Vert NC_i \Vert r_i^*)$		
		$M_{UG}^* = h(TID_i \Vert HID_i^* \Vert PID_i^{old} \Vert R_i^* \Vert T_1)$		
		Check $M_{UG}^* ? = M_{UG}$		
		Generate $tPID_i^{new}$		
		Set $PID_i^{new} = tPID_i^{new}$		
		Generate T_2		
		Update $UserinfoTable$, let $NG_j = NG_j + n_j, X_{S_j} = h(SID_j \Vert X_{S_j})$		
		Compute $M_2 = (R_i^* \Vert PID_i^{old}) \oplus H(X_{S_j} \Vert T_2)$		
		$M_{GS} = h(PID_i^{old} \Vert SID_j \Vert X_{S_j} \Vert R_i^* \Vert T_2)$		
			$\{M_2, M_{GS}, NG_j, T_2\} \rightarrow$	
			Check the validity of T_2	
			Compute $N' = (NG_j - N_j)/n_j$	
			Check if $1 \le N' \le N$	
			Let $X_{S_j}' = X_{S_j}$	
			Compute N' times $X_{S_j}' = h(X_{S_j}' \Vert SID_j)$	
			Compute $(R_i^* \Vert PID_i^{old}) = M_2 \oplus H(X_{S_j}' \Vert T_2)$	
			$M_{GS}^* = h(PID_i^{old} \Vert SID_j \Vert X_{S_j}' \Vert R_i^* \Vert T_2)$	
			Check $M_{GS}^* ? = M_{GS}$	
			Generate r_j, T_3	
			Compute $R_j = h(SID_j \Vert r_j)$	
			$M_3 = (R_j \Vert PID_i^{old}) \oplus H(X_{S_j}' \Vert T_3)$	
			$SK_{ji} = h(R_i^* \Vert R_j)$	
			$M_{SG} = h(PID_i^{old} \Vert SID_j \Vert X_{S_j}' \Vert R_i^* \Vert R_j \Vert T_3)$	
			Update $X_{S_j} = X_{S_j}', Nj = NG_j$	
			$\{SID_j, M_3, M_{SG}, T_3\} \leftarrow$	
		Check the validity of T_3		
		Compute $(R_j^* \Vert PID_i^{old}) = M_3 \oplus H(X_{S_j} \Vert T_3)$		
		$M_{SG}^* = h(PID_i^{old} \Vert SID_j \Vert X_{S_j} \Vert R_i^* \Vert R_j^* \Vert T_3)$		
		Check $M_{SG}^* ? = M_{SG}$		
		Generate T_4		
		Compute $C_i^{new} = h(TID_i \Vert HID_i^*) \oplus PID_i^{new}, p_i^{new} = C_i^{new} \oplus HID_i^* \oplus T_4$		
		$M_4 = R_j^* \oplus K_i, M_{GU} = h(PID_i^{old} \Vert HID_i^* \Vert C_i^{new} \Vert R_i^* \Vert R_j^* \Vert T_4)$		
	$\{p_i^{new}, M_4, M_{GU}, T_4\} \leftarrow$			
Check the validity of T_4				
Compute $R_j^* = M_4 \oplus H(TID_i \Vert HID_i^* \Vert NC_i \Vert T_1), SK_{ij} = h(R_i \Vert R_j^*)$				
$C_i^{new} = p_i^{new} \oplus HID_i^* \oplus T_4, M_{GU}^* = h(PID_i \Vert HID_i^* \Vert C_i^{nevv} \Vert R_i \Vert R_j^* \Vert T_4)$				
Check $M_{GU}^* ? = M_{GU}$				
Update $C_i = C_i^{new}$, $flag = 0$				

Figure 5. Authentication phase of the proposed scheme.

5.5. Password and Biometric Update Phase

Step 1: U_i inserts SC_i, inputs ID_i and PW_i, and imprints Bio_i. SC_i calculates $b_i = REP(Bio_i, pair_i)$, $TID_i = h(ID_i)$, $HPW_i^* = h(PW_i||b_i)$, $HID_i^* = A_i \bigoplus h(HPW_i^*||TID_i)$, and $B_i^* = h(HPW_i^*||HID_i^*)$ *mod n* and verifies the equality check for B_i^* ? $= B_i$. If it does not hold true, SC_i rejects the request. Otherwise, U_i inputs a new password PW_i^{new} and imprints a new biometric Bio_i^{new}.

Step 2: SC_i calculates $GEN(Bio_i^{new}) = (b_i^{new}, pair_i^{new})$, $HPW_i^{new} = h(PW_i^{new}||b_i^{new})$, $A_i^{new} = HID_i^* \bigoplus h(HPW_i^{new}||TID_i)$, and $B_i^{new} = h(HPW_i^{new}||HID_i^*)$ *mod n*. At last, SC_i replaces $pair_i$, A_i and B_i, with $pair_i^{new}$, A_i^{new} and B_i^{new}, respectively.

6. Security Analysis

6.1. Security Verification Using ProVerif

ProVerif [33] is one of the widely accepted automated security verification tools for communication protocols. ProVerif supports main cryptographic primitives including hash function, encryption, digital signatures, etc. In this section, we use ProVerif to check the mutual authentication and session key secrecy of the proposed scheme.

First, we define two insecure channels, c1 for communication between users and the *GWN* and c2 for communication between the *GWN* and sensors.

```
(*–Two public channel–*)
free c1: channel. (*–The channel between users and GWN–*)
free c2: channel. (*–The channel between sensors and GWN–*)
```

Then, we define the parameters and constructors as follows:

```
(*–The basic variables–*)
free user, GWN, SN: bitstring. (*—three participants–*)
free PID: bitstring. (*—the pseudonym identity shared by user and GWN–*)
free Ku: bitstring[private]. (*—the masterkey of GWN–*)
free Ks: bitstring[private]. (*—the masterkey of GWN–*)
free XSj: bitstring[private]. (*—the shared key between GWN and sensor–*)
table Table_user_info( bitstring, bitstring, bitstring). (*—the user's info table—*)
table Table_sensor_info(bitstring, bitstring). (*—the sensor's info table—*)
(*–Encryption operation–*)
fun encrypt(bitstring, bitstring): bitstring.
fun decrypt(bitstring, bitstring): bitstring.
equation forall x: bitstring, y: bitstring; decrypt(encrypt(x, y), y) = x.
(*–Hash operation–*)
fun h1(bitstring): bitstring.
fun h2 (bitstring, bitstring): bitstring.
fun h4 (bitstring, bitstring, bitstring, bitstring):bitstring.
fun h5 (bitstring, bitstring, bitstring, bitstring, bitstring): bitstring.
fun h6 (bitstring, bitstring, bitstring, bitstring, bitstring, bitstring): bitstring.
(*–XOR operation–*)
fun XOR (bitstring, bitstring): bitstring.
equation forall x: bitstring, y: bitstring; XOR(XOR(x, y), y) = x.
(*–Concat and Divide operation–*)
fun Concat (bitstring, bitstring): bitstring.
fun Div1 (bitstring):bitstring.
fun Div2 (bitstring):bitstring.
equation forall x: bitstring, y: bitstring; Div1(Concat(x, y)) = x.
equation forall x: bitstring, y: bitstring; Div2(Concat(x, y)) = y.
(*–Check the Freshness of timestamp operation–*)
fun isFresh (bitstring, bool): bool
reduc forall T: bitstring; isFresh (T, true) = true.
```

In order to check the mutual authentication and session key secrecy, we define the following eight events and two secrets:

```
(*–Mutual authentication queries–*)
event beginUG(bitstring).
event endUG(bitstring).
event beginGU(bitstring).
event endGU(bitstring).
event beginGS(bitstring).
event endGS(bitstring).
event beginSG(bitstring).
event endSG(bitstring).
query x: bitstring; inj-event(endUG(x)) ==> inj-event(beginUG(x)).
query x: bitstring; inj-event(endGU(x)) ==> inj-event(beginGU(x)).
query x: bitstring; inj-event(endGS(x)) ==> inj-event(beginGS(x)).
query x: bitstring; inj-event(endSG(x)) ==> inj-event(beginSG(x)).
(*–Session key secrecy queries–*)
free secretA, secretB: bitstring [private].
query attacker(secretA);
attacker(secretB).
```

Three distinct processes *processUser*, *processSensor*, and *processGWN* are declared to model U_i, S_j and *GWN*, respectively.

```
(*————————-User Ui————————————*)
let processUser(IDi: bitstring, PWi: bitstring, bi: bitstring, Ai: bitstring, Bi: bitstring,
Ci: bitstring, NCi: bitstring, SIDj: bitstring) =
let TIDi = h1(IDi) in
let HPWi' = h2(PWi, bi) in
let HIDi' = XOR(Ai, h2(HPWi', TIDi)) in
let Bi' = h2(HPWi',HIDi') in
if Bi'=Bi then
event beginGU(GWN);
new ri: bitstring;
new T1: bitstring;
let PIDi = XOR(Ci, h2(TIDi, HIDi')) in
let Ri = h4(TIDi, PIDi, NCi, ri) in
let M1 = XOR(Concat(ri, SIDj), h4 TIDi, HIDi', NCi, T1)) in
let MUG = h5 (TIDi, HIDi', PIDi, Ri, T1) in
out (c1, (PIDi, M1, MUG, T1));
in (c1, (M4:bitstring, MGU:bitstring, T4:bitstring));
if isFresh (T4, true) = true then
let Rj' = XOR(M4,h4(TIDi, HIDi',NCi, T1)) in
let SKij' = h2(Ri, Rj') in
let MGU' = h5(PIDi, HIDi', Ri, Rj', T4) in
if MGU' =MGU then
event endUG(user);
out(c1, encrypt(secretA, SKij')).
(*————————-GWN————————————*)
let processGWN() =
in(c1, (PIDi: bitstring, M1: bitstring, MUG: bitstring, T1:bitstring));
if isFresh(T1, true) = true then
get Table_user_info(=PIDi, TIDi, NCi) in
let HIDi' = h2(TIDi, Ku) in
let Ki = h4(TIDi, HIDi', NCi, T1) in
let ri' = Div1(XOR(M1,Ki)) in
let SIDj = Div2(XOR(M1,Ki)) in
let Ri' = h4(TIDi, PIDi, NCi, ri') in
event beginUG(user);
```

```
let MUG' = h5(TIDi, HIDi', PIDi, Ri', T1) in
if MUG' =MUG then
event beginSG(SN);
new T2: bitstring;
get Table_sensor_info(=SIDj, XSj) in
let M2 = XOR(Concat(Ri', PIDi),h2(XSj, T2)) in
let MGS = h5(PIDi, SIDj, XSj, Ri', T2) in
out(c2, (M2, MGS, T2));
in(c2,(SIDj:bitstring, M3:bitstring, MSG:bitstring, T3: bitstring));
if isFresh(T3, true) = true then
let Rj' = Div1(XOR(M3,h2(XSj, T3))) in
let PIDi = Div2(XOR(M3, h2(XSj, T3))) in
let MSG' = h6(PIDi, SIDj, XSj, Ri',Rj', T3) in
if MSG' = MSG then
new T4: bitstring;
event endGS(GWN);
let M4 = XOR(Rj',Ki) in
let MGU = h5(PIDi, HIDi', Ri', Rj', T4) in
out(c1, (M4, MGU, T4));
event endGU(GWN);
0.
(*————————-Sensor Sj————————————*)
let processSensor(SIDj:bitstring, XSj:bitstring) =
in(c2, (MG:bitstring, MGS:bitstring, T2:bitstring));
event beginGS(GWN);
let Ri' =Div1(XOR(MG, h2(XSj, T2))) in
let PIDi =Div2(XOR(MG, h2(XSj, T2))) in
let MGS' = h5(PIDi, SIDj, XSj, Ri', T2) in
if MGS'=MGS then
new T3: bitstring;
new rj: bitstring;
let Rj = h2(SIDj, rj) in
let Mj = XOR(Concat(rj, PIDi),h2(XSj, T3)) in
let SKij = h2(Ri', Rj) in
let MSG = h6(PIDi, SIDj, XSj, Ri', Rj, T3) in
out(c2, (SIDj, Mj, MSG, T3));
event endSG(SN);
out(c2, encrypt(secretB, SKij)).
```

We simulate the unbounded parallel execution of processes *processUser*, *processSensor*, and *processGWN* as follows:

```
(*–Start process–*)
process
new IDi: bitstring;
new PWi: bitstring;
new bi: bitstring;
new PIDi: bitstring;
new SIDj: bitstring;
new NCi: bitstring;
let HPWi = h2(PWi, bi) in
let TIDi = h1(IDi) in
let HIDi = h2(TIDi, Ku) in
let Ai = XOR(h2(HPWi, TIDi), HIDi) in
let Bi = h2(HPWi, HIDi) in
let Ci = XOR(h2(TIDi, HIDi),PIDi) in
```

```
let XSj = h2(SIDj, Ks) in
insert Table_user_info(PIDi, TIDi, NCi);
insert Table_sensor_info(SIDj, XSj);
(
(*– Launch an unbounded number of sessions of the user –*)
(!processUser(IDi, PWi, bi, Ai, Bi, Ci, NCi, SIDj)) |
(*– Launch an unbounded number of, sessions of the GWN–*)
(!processGWN()) |
(*– Launch an unbounded number of sessions of the sensor–*)
(!processSensor(SIDj, XSj))
)
```

The simulation results are shown as follows:

Query inj-event(endUG(x)) ==> inj-event(beginUG(x)) is true.
Query inj-event(endGU(x)) ==> inj-event(beginGU(x)) is true.
Query inj-event(endGS(x)) ==> inj-event(beginGS(x)) is true.
Query inj-event(endSG(x)) ==> inj-event(beginSG(x)) is true.
Query not attacker(secretA[]) is true.
Query not attacker(secretB[]) is true.

The results mean that the proposed scheme is able to achieve mutual authentication. Meanwhile, the session key *SKij* generated by the user U_i and the sensor S_j is secure.

6.2. BAN-Logic

Burrows–Abadi–Needham logic (*BAN*-logic) [34] is a widely used tool for the formal analysis of authentication schemes which was proposed by Burrows et al. In this section, we use *BAN*-logic to prove the session key agreement between the user U_i and the sensor node S_j after the execution of the proposed scheme. Table 3 introduces the notations for the *BAN*-logic analysis and some basic rules for *BAN*-logic are described in Table 4.

Table 3. *BAN*-logic notations.

Symbol	Description
$P\mid\equiv X$	P believes X.
$P \lhd X$	P sees X.
$P\mid\sim X$	P sends X.
$P \Rightarrow X$	P has jurisdiction over X.
(X)	X is fresh.
$(X,\ Y)$	X or Y is part of $(X,\ Y)$.
$(X)_K$	Use the key K to compute X.
$P \overset{SK}{\leftrightarrow} Q$	P and Q achieve the shared key SK for communication.

Table 4. Basic logical postulates of *BAN*-logic.

Symbol	Description
Message meaning rule	$\frac{P\mid\equiv(P \overset{K}{\leftrightarrow} Q),\ P\lhd(X)_K}{P\mid\equiv Q\mid\sim X}$
Freshness conjuncatenation rule	$\frac{P\mid\equiv(X)}{P\mid\equiv(X,\ Y)}$
Nonce verification rule	$\frac{P\mid\equiv(X),\ P\mid\equiv Q\mid\sim X}{P\mid\equiv Q\mid\equiv X}$
Jurisdiction rule	$\frac{P\mid\equiv Q\mid\Rightarrow X,\ P\mid\equiv Q\mid\equiv X}{P\mid\equiv X}$
Believe rule	$\frac{P\mid\equiv Q\mid\equiv(X,\ Y)}{P\mid\equiv Q\mid\equiv X}$, $\frac{P\mid\equiv X, P\mid\equiv Y}{P\mid\equiv(X,\ Y)}$

(1) The idealized form of the proposed scheme:
Message 1: $U_i \rightarrow GWN:\ (r_i,\ SID_j)_{U_i \overset{(HID_i,\ NC_i)}{\leftrightarrow} GWN}$

Message 2: $GWN \rightarrow S_j : (R_i, PID_i^{old})_{GWN \overset{X_{S_j}}{\leftrightarrow} S_j}$

Message 3: $S_j \rightarrow GWN : (R_j, PID_i^{old})_{S_j \overset{X_{S_j}}{\leftrightarrow} GWN}$

Message 4: $GWN \rightarrow U_i : (R_j)_{GWN \overset{(HID_i', NC_i)}{\leftrightarrow} U_i}$

(2) Verification goals:

Goal 1: $U_i| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$.

Goal 2: $U_i| \equiv S_j| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$.

Goal 3: $S_j| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$.

Goal 4: $S_j| \equiv U_i| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$.

(3) Assumptions about the initial state:

A1: $U_i| \equiv (r_i, r_j)$.

A2: $S_j| \equiv (r_i, r_j)$.

A3: $GWN| \equiv (r_i, r_j)$.

A4: $U_i| \equiv (U_i \overset{(PID_i, HID_i, NC_i)}{\leftrightarrow} GWN)$.

A5: $GWN| \equiv (U_i \overset{(PID_i, HID_i, NC_i)}{\leftrightarrow} GWN)$.

A6: $GWN| \equiv (GWN \overset{X_{S_j}}{\leftrightarrow} S_j)$.

A7: $S_j| \equiv (GWN \overset{X_{S_j}}{\leftrightarrow} S_j)$.

A8: $U_i| \equiv S_j \Rightarrow (U_i \overset{SK}{\leftrightarrow} S_j)$.

A9: $S_j| \equiv U_i \Rightarrow (U_i \overset{SK}{\leftrightarrow} S_j)$

(4) Proofs:

Step 1: From Message 1, we can get: $GWN \lhd (r_i, SID_j)_{U_i \overset{(HID_i', NC_i)}{\leftrightarrow} GWN}$.

Step 2: According to Step 1, A5, and the message meaning rule, it can be inferred that: $GWN| \equiv U_i| \sim (r_i, SID_j)_{U_i \overset{(HID_i', NC_i)}{\leftrightarrow} GWN}$.

Step 3: According to Step 2, A3, and the nonce verification rule, we obtain: $GWN| \equiv U_i| \equiv (r_i, SID_j)_{U_i \overset{(HID_i', NC_i)}{\leftrightarrow} GWN}$.

Step 4: From Message 2, we understand that: $S_j \lhd (R_i, PID_i^{old})_{GWN \overset{X_{S_j}}{\leftrightarrow} S_j}$.

Step 5: According to A7 and the message meaning rule, we obtain: $S_j| \equiv GWN| \sim (R_i, PID_i^{old})_{GWN \overset{X_{S_j}}{\leftrightarrow} S_j}$.

Step 6: According to A2, $R_i = h(TID_i||PID_i||NC_i||r_i)$, and the freshness conjuncatenation rule, we can get: $S_j| \equiv (R_i)$.

Step 7: According to Step 5, Step 6, and the nonce verification rule, we get: $S_j| \equiv GWN| \equiv (R_i, PID_i^{old})_{GWN \overset{X_{S_j}}{\leftrightarrow} S_j}$.

Step 8: According to Step 3, Step 7, $SK_{ij} = h(R_i||R_j)$, and $R_i = h(TID_i||PID_i||NC_i||r_i)$, we prove: $S_j| \equiv U_i| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$ (**Goal 4**).

Step 9: According to Step 8, A9, and the jurisdiction rule, we prove: $S_j| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$ (**Goal 3**).

Step 10: According to Message 3, we get: $GWN \lhd (R_j, PID_i^{old})_{S_j \overset{X_{S_j}}{\leftrightarrow} GWN}$.

Step 11: According to Step10, A6, and the message meaning rule, it can be inferred that: $GWN| \equiv S_j| \sim (R_j, PID_i^{old})_{S_j \overset{X_{S_j}}{\leftrightarrow} GWN}$.

Step 12: According to Step 11, A3, $R_j = h(SID_j||r_j)$, and the nonce verification rule, we obtain: $GWN| \equiv S_j| \equiv (R_j, PID_i^{old}) \underset{S_j \leftrightarrow GWN}{\overset{X_{S_j}}{}}$.

Step 13: From Message 4, we obtain: $U_i \lhd (R_j)_{GWN^{(HID_i', NC_i)}_{U_i}}$.

Step 14: According to Step 13, A4, and the message meaning rule, we obtain: $U_i| \equiv GWN|\sim(R_j)_{GWN^{(HID_i', NC_i)}_{U_i}}$.

Step 15 According to Step 14, A1, $R_j = h(SID_j||r_j)$, and the nonce verification rule, we get: $U_i| \equiv S_j| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$. (**Goal 2**).

Step 16: According to Step 15, A8, and the jurisdiction rule, we prove: $U_i| \equiv (U_i \overset{SK}{\leftrightarrow} S_j)$ (**Goal 1**).

From the proof results obtained from the above process, **Goal 1–4**, U_i, and S_j believe that they have completed the key agreement and generated the shared session key $SK_{ij} = SK_{jiji}$.

6.3. Informal Security Analysis

(1) Anonymity and un-traceability

Suppose an adversary intercepted the information transmitted to a public channel from U_i, *GWN*, and S_j. Obviously, the adversary cannot obtain the user's actual identity ID_i, because of the security of the one-way *hash* function. In addition, the pseudonym identity PID_i changes after each authentication, and r_i and r_j are randomly generated in each session. The adversary cannot determine whether two sessions are launched by the same user.

(2) Perfect forward secrecy

Suppose an adversary accidentally captured U_i's private key NC_i, S_j's private key (X_{S_j}, N_j), and the *GWN*'s master key (K_u, K_s), and intercepted the information propagated in the public channel. The adversary cannot obtain the previous session key because NC_i, X_{S_j}, N_j changes after each authentication, and the adversary cannot get the NC_i, X_{S_j}, N_j in a previous session because of the security of the one-way *hash* function. Therefore, the proposed scheme can achieve perfect forward secrecy.

(3) Mutual authentication

In Section 6.1, we define eight events—event *beginUGparam*(bitstring), event *endUGparam*(bitstring), event *beginGUparam*(bitstring), event *endGUparam*(bitstring), event *beginGSparam*(bitstring), event *endGSparam*(bitstring), event *beginSGparam*(bitstring), and event *endSGparam*(bitstring)—to verify the mutual authentication of U_i, *GWN*, and S_j. The results show that our proposed scheme could achieve mutual authentication.

(4) Session key agreement

The user U_i and the sensor S_j reach a session key $SK_{ij} = h_2(R_i||R_j)$ for future communication after authentication. Since R_i and R_j are generated by U_i and S_j, respectively, both U_i and S_j have an influence on the outcome of the session key $SK_{ij} = h(R_i||R_j) = SK_{ji}$.

(5) Three-factor security

Suppose an adversary captured the smart card SC_i of U_i and obtained the biometrics Bio_i. The adversary can extract the values $\{A_i, b_i, C_i, NC_i, pair_i, flag\}$ in SC_i. Further, the adversary guesses (ID_i', PW_i') and calculates $b_i = REP(Bio_i, pair_i)$, $TID_i' = h(ID_i')$, $HPW_i^* = h(PW_i'||b_i)$, $HID_i^* = A_i \bigoplus h(HPW_i^*||TID_i')$, and $b_i^* = h(HPW_i^*||HID_i^*)\ mod\ n$. However, the adversary does not know the correctness of (ID_i', PW_i') because $B_i\ ? = h(HPW_i^*||HID_i^*)\ mod\ n$. is a fuzzy verification process.

(6) Resistance of other known attacks

Insider attack: An insider adversary can obtain the user's registration information $\{TID_i = h(ID_i),\ HPW_i = h(PW_i||b_i)\}$. Because of the security of the one-way hash function and ignorance about b_i, the adversary cannot capture PW_i. Therefore, no effective insider attack can be launched.

Stolen verifier table attack: There is no password-related or biometric-related information stored inside the *GWN*. Therefore, the stolen verifier table attack is infeasible in our proposed scheme.

User impersonation attack: For generating valid login request information, the adversary needs to know HID_i. While we know $b_i = REP(Bio_i, pair_i)$, $TID_i = h(ID_i)$, $HPW_i = h(PW_i||b_i)$, and $HID_i = A_i \bigoplus h(HPW_i||TID_i)$, where A_i and $pair_i$ are stored in SC_i. Therefore, the adversary cannot forge U_i without getting ID_i, PW_i, Bio_i and SC_i. Thus our proposed scheme could resist user impersonation attacks.

Sensor Spoofing Attack: An adversary cannot forge a sensor node S_j without getting the secrets of S_j (NS_j and X_{S_j}). Therefore, no effective sensor spoofing attack can be launched.

Known session-specific temporary information attack: In our proposed scheme, the user U_i and the micro-sensor S_j reach a session key $SK_{ij} = h(R_i||R_j) = h(h(TID_i||PID_i||NC_i||r_i)||h(SID_j||r_j))$. Even if an adversary captured the session-specific temporary information, r_i and r_j, he cannot launch a known session-specific temporary information attack without NC_i. As a result, our proposed scheme can resist known session-specific temporary information attacks.

De-synchronization attack: We analyze five possible cases of de-synchronization attacks, shown in Figure 6.

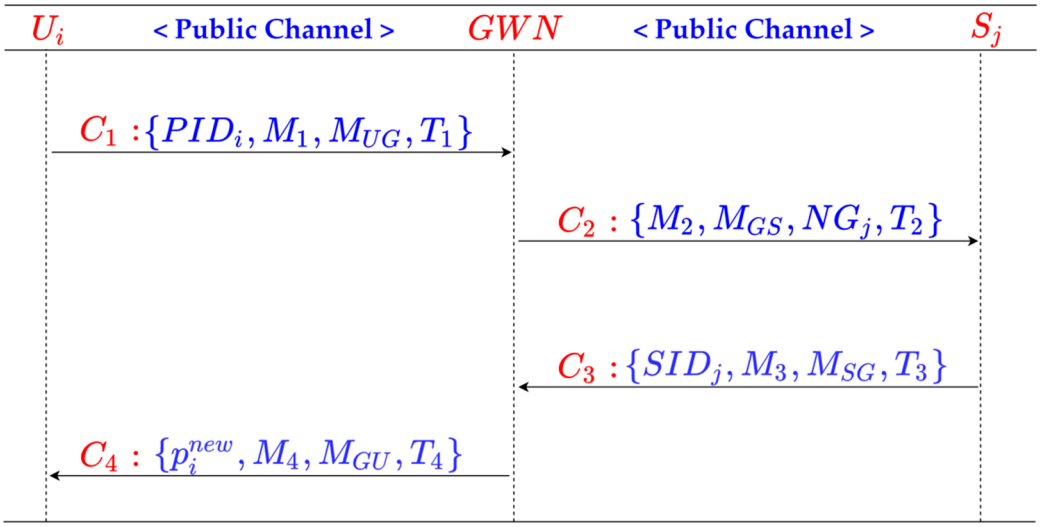

Figure 6. Possible de-synchronization attack on our proposed protocol.

Case 1: Suppose an adversary blocked $C1 : \{PID_i, M_1, M_{UG}, T_1\}$. Since none of the participants updated the information table, the attack is infeasible.

Case 2: Suppose an adversary blocked $C2 : \{M_2, M_{GS}, NG_j, T_2\}$, the information stored on the *GWN* side and sensor side would be out of synchronization. However, by calculating

, it can be known how many times the communications between S_j and the *GWN* are blocked. The information on two sides would be resynchronized by calculating N_i times $X'_{S_j} = h(X'_{S_j}||SID_j)$ and updating $X_{S_j} = X'_{S_j}$, $N_j = NG_j$.

Case 3: If an adversary blocks $C3 : \{SID_j, M_3, M_{SG}, T_3\}$. Both the *GWN* and S_j have updated the X_{S_j} and NG_j/N_j. The synchronization between U_i and the *GWN* is the same as Case 4.

Case 4: If an adversary blocks $C4: \{p_i^{new}, M_4, M_{\mathrm{GU}}, T_4\}$. The communications between the *GWN* and S_j are in synchronization, while the communications between U_i and the *GWN* are out of synchronization. In this case, the *GWN* has completed the update of PID_i, and PID_i^{old} records the previous PID_i. Since $C4: \{p_i^{new}, M_4, M_{\mathrm{GU}}, T_4\}$ is not received, the user U_i does not update C_i, and the PID_i calculated in the next session is not updated. However, when U_i initiates the session request again, the *GWN* finds that the PID_i sent by U_i is the same as the PID_i^{old} recorded in its memory. The *GWN* can identify the de-synchronization attack initiated by the adversary and synchronize the information according to Step 2 of the authentication phase. Therefore, the proposed new protocol can also resist the attacks of Case 3 and Case 4 of de-synchronization attacks.

In summary, our proposed scheme can resist de-synchronization attacks. On the other hand, we have shown that our proposed scheme can achieve forward secrecy in previous part of this section. Therefore, our proposed scheme can resist offline password-guessing attacks and stolen smart card attacks.

7. Performance Analysis

This section will compare and analyze the performance of the proposed new protocol with other similar protocols, including a computing cost comparison and communication cost comparison. Since the registration phase of users and sensors occurs only once, and users do not change their passwords and biometrics frequently, this section only discusses the performance comparison between authentication phases.

7.1. Comparison of Computing Costs

According to the experimental data in the literature [35], $T_h \approx 0.32$ms, the computing cost comparison between our proposed scheme and other similar schemes, is shown in Table 5. From the results, the proposed protocol has a lower computation cost than the other four similar protocols.

Table 5. Comparison of computing costs (milliseconds).

Protocol	User	GWN	Sensor	Total
Shin et al. [26]	$13T_h \approx 4.16$	$15T_h \approx 4.8$	$6T_h \approx 1.92$	$34T_h \approx 10.88$
Ostad et al. [36]	$11T_h \approx 3.52$	$17T_h \approx 5.44$	$5T_h \approx 1.6$	$33T_h \approx 10.56$
Wu et al. [13]	$10T_h \approx 3.2$	$15T_h \approx 4.8$	$5T_h \approx 1.6$	$31T_h \approx 9.92$
Amin et al. [24]	$12T_h \approx 3.84$	$15T_h \approx 3.2$	$5T_h \approx 1.6$	$32T_h \approx 10.24$
Proposed	$11T_h \approx 3.52$	$10T_h \approx 3.2$	$6T_h \approx 1.92$	$27T_h \approx 8.64$

7.2. Comparison of Communication Costs

We assume that the length of identification, random number, timestamp, and other parameters involved in the proposed protocol and other similar protocols is 128 bits, and the length of the timestamp is 32 bits. *Hash* functions $h: \{0,1\}^* \bigoplus \{0, 1\}^{128}$ and $H: \{0, 1\}^* \bigoplus \{0, 1\}^{256}$ have 128-bit and 256-bit outputs, respectively. Other related protocols use hash functions (such as *MD*5) with an output length of 128 bits.

In the authentication phase of the newly proposed protocol, there are four transmission messages: $\{PID_i, M_1, M_{UG}, T_1\}$, $\{M_2, M_{\mathrm{GS}}, NG_j, T_2\}$, $\{SID_j, M_3, M_{\mathrm{SG}}, T_3\}$, and $\{p_i^{new}, M_4, M_{\mathrm{GU}}, T_4\}$. The total length of the transmitted message is (128 + 256 + 128 + 32) + (256 + 128 + 128 + 32) + (128 + 256 + 128 + 32) + (128 + 128 + 128 + 32) = 2048 bits.

Table 6 shows the comparison of the communication costs between the proposed new protocol and other similar schemes. From the comparison results, our new proposed scheme is also at a good level in terms of communication costs.

Table 6. Comparison of communication costs.

Protocol	Number of Messages	Length of Interactive Information
Shin et al. [26]	4 Messages	1664 bits
Ostad et al. [36]	6 Messages	2208 bits
Wu et al. [13]	4 Messages	2176 bits
Amin et al. [24]	6 Messages	2016 bits
Proposed	4 Messages	2048 bits

8. Conclusions

Due to the insecurity of wireless sensor networks, abundant research on authentication and key agreement protocols for *WSN*s has been put forward. In 2019, Shin et al. proposed a lightweight three-factor authentication and key agreement protocol based on symmetric cryptographic primitives for *WSN*s, which looked promising. However, we found that there are some security risks in their protocol. To solve the shortcomings, we proposed a new lightweight and anonymous three-factor authentication scheme for *WSN*s. Furthermore, we proved that our proposed scheme is secure using the automated security verification tool *ProVerif*, *BAN*-logic verification, and an informal security analysis. Through a performance comparison and analysis, our new scheme shows a good level of computing and communication overhead and has high practicability. In future research, we will focus on finding a lighter mathematical model to realize the strong security of identity authentication in wireless sensor networks and apply the scheme to the actual environment.

Author Contributions: Conceptualization, L.Z.; Formal analysis, H.X.; Investigation, H.X.; Methodology, L.Z.; Supervision, H.X. and K.Z.; Validation, K.Z.; Writing—original draft, L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Project of Educational Commission of Guangdong Province (No.6019210033S), Shenzhen Fundamental Research Project (No.JCYJ20180305163701198), Shenzhen Bureau of Education (No.zdzz20002).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless sensor network survey. *Comput. Netw.* **2008**, *52*, 2292–2330. [CrossRef]
2. Gnawali, O.; Jang, K.-Y.; Paek, J.; Vieira, M.; Govindan, R.; Greenstein, B.; Joki, A.; Estrin, D.; Kohler, E. The tenet architecture for tiered sensor networks. In Proceedings of the 4th International Conference on Embedded Networked Sensor Systems ACM, Boulder, CO, USA, 31 October–3 November 2006; pp. 153–166.
3. Yang, D.; Misra, S.; Fang, X.; Xue, G.; Zhang, J. Two-Tiered Constrained Relay Node Placement in Wireless Sensor Networks: Computational Complexity and Efficient Approximations. *IEEE Trans. Mob. Comput.* **2011**, *11*, 1399–1411. [CrossRef]
4. He, D.; Kumar, N.; Chilamkurti, N. A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks. *Inf. Sci.* **2015**, *321*, 263–277. [CrossRef]
5. He, D.; Chen, C.; Chan, S.; Bu, J.; Yang, L.T. Security Analysis and Improvement of a Secure and Distributed Reprogramming Protocol for Wireless Sensor Networks. *IEEE Trans. Ind. Electron.* **2012**, *60*, 5348–5354. [CrossRef]
6. Lamport, L. Password authentication with insecure communication. *Commun. ACM* **1981**, *24*, 770–772. [CrossRef]
7. Guo, H.; Gao, Y.; Xu, T.; Zhang, X.; Ye, J. A secure and efficient three-factor multi-gateway authentication protocol for wireless sensor networks. *Ad Hoc Networks* **2019**, *95*, 101965. [CrossRef]
8. Gope, P.; Das, A.K.; Kumar, N.; Cheng, Y. Lightweight and Physically Secure Anonymous Mutual Authentication Protocol for Real-Time Data Access in Industrial Wireless Sensor Networks. *IEEE Trans. Ind. Inform.* **2019**, *15*, 4957–4968. [CrossRef]
9. Ever, Y.K. Secure-anonymous user Authentication scheme for e-healthcare application using wireless medical sensor networks. *IEEE Syst. J.* **2018**, *13*, 456–467. [CrossRef]

10. Adavoudi-Jolfaei, A.H.; Ashouri-Talouki, M.; Aghili, S.F. Lightweight and anonymous three-factor authentication and access control scheme for real-time applications in wireless sensor networks. *Peer-to-Peer Netw. Appl.* **2019**, *12*, 43–59. [CrossRef]
11. Wang, D.; Li, W.; Wang, P. Measuring Two-Factor Authentication Schemes for Real-Time Data Access in Industrial Wireless Sensor Networks. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4081–4092. [CrossRef]
12. Mishra, D.; Vijayakumar, P.; Sureshkumar, V.; Amin, R.; Islam, S.K.H.; Gope, P. Efficient authentication protocol for secure multimedia communications in IoT-enabled wireless sensor networks. *Multimed. Tools Appl.* **2018**, *77*, 18295–18325. [CrossRef]
13. Wu, F.; Li, X.; Sangaiah, A.K.; Xu, L.; Kumari, S.; Wu, L.; Shen, J. A lightweight and robust two-factor authentication scheme for personalized healthcare systems using wireless medical sensor networks. *Futur. Gener. Comput. Syst.* **2018**, *82*, 727–737. [CrossRef]
14. Wu, F.; Xu, L.; Kumari, S.; Li, X. An improved and provably secure three-factor user authentication scheme for wireless sensor networks. *Peer-to-Peer Netw. Appl.* **2018**, *11*, 1–20. [CrossRef]
15. Ramachandran, S.; Shanmugam, V. A two way authentication using bilinear mapping function for wireless sensor networks. *Comput. Electr. Eng.* **2017**, *59*, 242–249. [CrossRef]
16. Gope, P.; Hwang, T. A Realistic Lightweight Anonymous Authentication Protocol for Securing Real-Time Application Data Access in Wireless Sensor Networks. *IEEE Trans. Ind. Electron.* **2016**, *63*, 7124–7132. [CrossRef]
17. Kumari, S.; Li, X.; Wu, F.; Das, A.K.; Arshad, H.; Khan, M.K. A user friendly mutual authentication and key agreement scheme for wireless sensor networks using chaotic maps. *Futur. Gener. Comput. Syst.* **2016**, *63*, 56–75. [CrossRef]
18. Xiong, L.; Peng, D.; Peng, T.; Liang, H.; Liu, Z. A Lightweight Anonymous Authentication Protocol with Perfect Forward Secrecy for Wireless Sensor Networks. *Sensors* **2017**, *17*, 2681. [CrossRef]
19. Wong, K.H.M.; Zheng, Y.; Cao, J.; Wang, S. A dynamic user authentication scheme for wireless sensor networks. In Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC '06), Taichung, Taiwan, 5–7 June 2006; Volume 1, p. 8.
20. Das, M.L. Two-factor user authentication in wireless sensor networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1086–1090. [CrossRef]
21. Chen, T.H.; Shih, W.K. A robust mutual authentication protocol for wireless sensor networks. *ETRI J.* **2010**, *32*, 704–712. [CrossRef]
22. He, D.; Gao, Y.; Chan, S.; Chen, C.L.P. An enhanced two-factor user authentication scheme in wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* **2010**, *10*, 361–371.
23. Farash, M.S.; Turkanović, M.; Kumari, S.; Hölbl, M. An efficient user authentication and key agreement scheme for heterogeneous wireless sensor network tailored for the Internet of Things environment. *Ad Hoc Networks* **2016**, *36*, 152–176. [CrossRef]
24. Amin, R.; Islam, S.H.; Biswas, G.; Khan, M.K.; Leng, L.; Kumar, N. Design of an anonymity-preserving three-factor authenticated key exchange protocol for wireless sensor networks. *Comput. Networks* **2016**, *101*, 42–62. [CrossRef]
25. Jiang, Q.; Zeadally, S.; Ma, J.; He, D. Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks. *IEEE Access* **2017**, *5*, 3376–3392. [CrossRef]
26. Shin, S.; Kwon, T. A Lightweight Three-Factor Authentication and Key Agreement Scheme in Wireless Sensor Networks for Smart Homes. *Sensors* **2019**, *19*, 2012. [CrossRef]
27. Fathollahi-Fard, A.M.; Dulebenets, M.A.; Hajiaghaei–Keshteli, M.; Tavakkoli-Moghaddam, R.; Safaeian, M.; Mirzahosseinian, H. Two hybrid meta-heuristic algorithms for a dual-channel closed-loop supply chain network design problem in the tire industry under uncertainty. *Adv. Eng. Inform.* **2021**, *50*, 101418. [CrossRef]
28. Fathollahi-Fard, A.M.; Ahmadi, A.; Karimi, B. Multi-Objective Optimization of Home Healthcare with Working-Time Balancing and Care Continuity. *Sustainability* **2021**, *13*, 12431. [CrossRef]
29. Fallahpour, A.; Wong, K.Y.; Rajoo, S.; Fathollahi-Fard, A.M.; Antucheviciene, J.; Nayeri, S. An integrated approach for a sustainable supplier selection based on Industry 4.0 concept. *Environ. Sci. Pollut. Res.* **2021**, 1–19. [CrossRef]
30. Wang, D.; He, D.; Wang, P.; Chu, C.-H. Anonymous Two-Factor Authentication in Distributed Systems: Certain Goals Are Beyond Attainment. *IEEE Trans. Dependable Secur. Comput.* **2015**, *12*, 428–442. [CrossRef]
31. Huang, X.; Xiang, Y.; Chonka, A.; Zhou, J.; Deng, R. A Generic Framework for Three-Factor Authentication: Preserving Security and Privacy in Distributed Systems. *IEEE Trans. Parallel Distrib. Syst.* **2011**, *22*, 1390–1397. [CrossRef]
32. Wang, D.; Gu, Q.; Cheng, H.; Wang, P. The request for better measurement: A comparative evaluation of two-factor authentication schemes. In Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security ACM, Xi'an, China, 30 May–3 June 2016; pp. 475–486.
33. Blanchet, B.; Smyth, B.; Cheval, V.; Sylvestre, M. ProVerif 2.00: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial. 2018. Available online: https://prosecco.gforge.inria.fr/personal/bblanche/proverif (accessed on 15 October 2021).
34. Burrows, M.; Abadi, M.; Needham, R.M. A logic of authentication. Proceedings of the Royal Society of London. *A. Math. Phys. Sci.* **1989**, *426*, 233–271.
35. He, D.; Kumar, N.; Lee, J.-H.; Sherratt, R. Enhanced three-factor security protocol for consumer USB mass storage devices. *IEEE Trans. Consum. Electron.* **2014**, *60*, 30–37. [CrossRef]
36. Ostad-Sharif, A.; Arshad, H.; Nikooghadam, M.; Abbasinezhad-Mood, D. Three party secure data transmission in IoT networks through design of a lightweight authenticated key agreement scheme. *Futur. Gener. Comput. Syst.* **2019**, *100*, 882–892. [CrossRef]