Secure and Efficient Three-Factor Protocol for Wireless Sensor Networks

Jihyeon Ryu 1, Hakjun Lee 2, Hyoungshick Kim 3 and Dongho Won 3,*

1 Department of Platform Software, Sungkyunkwan University, Gyeonggi-do 16419, Korea; jhryu@security.re.kr
2 Department of Electrical and Computer Engineering, Sungkyunkwan University, Gyeonggi-do 16419, Korea; hjlee@security.re.kr
3 Department of Computer Engineering, Sungkyunkwan University, Gyeonggi-do 16419, Korea; hyoung@skku.edu
* Correspondence: dhwon@security.re.kr; Tel.: +82-31-290-7107

Received: 8 November 2018; Accepted: 14 December 2018; Published: 18 December 2018

Abstract: Wireless sensor networks are widely used in many applications such as environmental monitoring, health care, smart grid and surveillance. Many security protocols have been proposed and intensively studied due to the inherent nature of wireless networks. In particular, Wu et al. proposed a promising authentication scheme which is sufficiently robust against various attacks. However, according to our analysis, Wu et al.’s scheme has two serious security weaknesses against malicious outsiders. First, their scheme can lead to user impersonation attacks. Second, user anonymity is not preserved in their scheme. In this paper, we present these vulnerabilities of Wu et al.’s scheme in detail. We also propose a new scheme to complement their weaknesses. We improve and speed up the vulnerability of the Wu et al. scheme. Security analysis is analyzed by Proverif and informal analysis is performed for various attacks.

Keywords: wireless sensor networks; user authentication; biometric; smart card

1. Introduction

A wireless sensor network (WSN) is a distributed network of autonomous sensors that are typically used to collect information about environmental or physical conditions. Wireless sensor networks are applicable to a variety of applications such as environmental monitoring, health care, smart grid and surveillance [1–6] because they can be easily deployed without a significant cost penalty.

In general, a WSN system consists of four entities: (1) user interface, (2) a sensor node that measures physical or environmental conditions, (3) a gateway node that forwards the information received from the sensor nodes to a central server, and (4) a central server that collects the information from the sensor nodes and analyze it. Naturally, however, the security of WSN is critical because network packets can be easily captured and modified in WSN due to the inherent characteristics of wireless networks. Therefore, we need to provide security protocols in order to ensure security properties such as confidentiality, integrity, and authenticity even when data packets on a WSN are captured and modified in an unauthorized manner.

Due to the inherent weakness of WSNs, many researchers have proposed security protocols to achieve fundamental security goals of WSNs. As one of the pioneers in this area, Watro et al. [7] proposed a security protocol using RSA (See Table A1 for details) for wireless sensor networks. To enhance the security of the authentication procedure, Das [2] extended their protocol to a two-factor user authentication protocol for WSNs where a user has to hold both a password and smartcard. Because their proposed authentication scheme provides reasonable security properties, it has been
widely used for WSNs as a de-factor standard protocol [8–10]. However, He et al. [11] found that Das’s protocol is vulnerable to several attacks such as insider attacks, impersonation attacks and lack of secure mutual authentication. They also suggested an authentication scheme by fixing the discovered problems. However, Kumar et al. [12] also discovered several security flaws such as information leakage, no session key agreement, no mutual authentication, and lack of anonymity in Das’s protocol.

Recently, some researchers (e.g., [13]) have started to develop user authentication schemes for WSNs using ECC, which can provide the same security as RSA with a smaller key size. ECC is the most efficient algorithm that satisfies forward secrecy and backward secrecy among the algorithms so far. Xue et al. [14] particularly introduced a temporal-credential-based protocol to provide user anonymity. However, Jiang et al. [15] demonstrated that Xue et al.’s scheme has four critical security flaws: (1) identity guessing attacks, (2) online password guessing attacks by privileged insiders, and (3) offline password guessing attacks with a victim’s smartcard. Jiang et al. also suggested a new authentication scheme to address their discovered issues.

More recently, Das [16] found that Jiang et al. [15]’s scheme has significant security issues such as the vulnerabilities to insider and de-synchronization attacks and lack of formal security proof of the proposed scheme. To address these issues, Das proposed several three-factor user authentication schemes [16–18] by introducing a new factor of user biometrics. Again, Wu et al. [1] found that all the Das’ schemes [16–18] are vulnerable to de-synchronization and offline password guessing attacks. In addition, the protocols [17,18] are vulnerable to user impersonation and offline password guessing attacks. To fix such problems, Wu et al. [1] suggested a three-factor user authentication scheme using ECC for WSNs.

In this paper, however, we found that Wu et al.’s scheme [1] has two security flaws against outsider attackers. First, their scheme can lead to user impersonation attacks. Second, user anonymity is not preserved because the user identity can be revealed from an anonymous login request message. We will explain these in the reminder of this paper. Our key contributions are summarized below:

- We discovered two security weaknesses in Wu et al.’s scheme [1], which was recently designed for user authentication using ECC in WSN systems. We demonstrated that a malicious outsider holding a smart card can extract the secret parameters from his/her smart card; the extracted secret parameters can be used to perform impersonation attacks and reveal the identity of the user from a login request message.
- We also proposed a novel three-factor user authentication scheme for WSN by extending Wu et al.’s scheme [1]. The proposed authentication scheme not only accomplishes several important security properties but also improves the performance of the protocol in time.

The rest of the paper is structured as follows: Section 2 gives some preliminaries of the cryptographic primitives (i.e., ECC and fuzzy extractor) used in our paper and explains the threat model and assumptions. Section 3 provides a review of Wu et al.’s scheme [1]. Section 4 analyzes the security weaknesses of their scheme. Section 5 presents a novel three-factor user authentication scheme by fixing security issues in Wu et al.’s scheme. Sections 6 and 7 provide security and performance analysis results, respectively. We conclude in Section 8.

2. Preliminaries

In this section, we introduce elliptic curves, fuzzy extractors, and threat models to be used in this paper.
2.1. Elliptic Curve Cryptosystem

The Elliptic curve cryptosystem (ECC) is the most frequently used password system in modern passwords and has strong security characteristics. Miller [19] and Neal [20] create ECC in 1985 and 1987, respectively. ECC uses the following formula:

\[ y^2 = x^3 + ax + b \mod p, \quad a, b \in F_p. \]  

(1)

The above equation is ECC on the \( F_p \). The following conditions must be met in order to ensure safety:

\[ 4a^3 + 27b^2 \not\equiv 0 \mod p. \]  

(2)

This is a formula that guarantees the non-singularity of an elliptic curve. When using this elliptic curve, safety is ensured as follows:

1. **Elliptic Curve Computational Diffie–Hellman Problem (ECCDHP):** Given \( xyP \), it is impossible to find \( xP, yP \).
2. **Elliptic Curve Decisional Diffie–Hellman Problem (ECDDHP):** Given \( xP, yP \) it is impossible to find \( xyP \).
3. **Elliptic Curve Discrete Logarithm Problem (ECDLP):** Given \( P, xP \) it is impossible to find \( x \).

We hypothesized that \( P \) is the point on \( F_p \), \( xP \) is the result of calculating \( P \) times \( x \), \( yP \) is the result of calculating \( P \) times \( y \), and \( xyP \) is the result of calculating \( P \) times \( xy \).

2.2. Fuzzy Extractor

The user’s biometric information is very important information. In general, human biometric recognition is perceived differently each time, and the fuzzy extractor plays a role in correcting it. The fuzzy extractor can obtain a unique string using error tolerance. The fuzzy extractor is operated through two procedures (\( \text{Gen} \), \( \text{Rep} \)), demonstrated as [17,21]:

\[ \text{Gen} (B) \rightarrow \langle \alpha, \beta \rangle, \]  

(3)

\[ \text{Rep} (B^*, \beta) = \alpha. \]  

(4)

\( \text{Gen} \) is a probabilistic generation function for which the biometrics \( B \) returns a factored out string \( \alpha \in \{0, 1\}^k \) and a coadjuvant string \( \beta \in \{0, 1\}^* \), and \( \text{Rep} \) is a function that restores \( \beta \) to \( \alpha \), and any vector \( B^* \) close to \( B \) [22].

2.3. Threat Assumption

We introduce a threat model [8], and consider constructing the threat assumptions as follows:

1. The attacker \( A \) can be a user, a gateway, or a sensor. Any registered user can act as an attacker.
2. \( A \) can intercept or eavesdrop on all communication messages in a public channel, thereby capturing any message exchanged between a user and gateway or sensor.
3. \( A \) has the ability to modify, reroute, or delete the intercepted message.
4. Stored parameters can be extracted from smart cards using the side channel attack [23].
5. An external attacker \( A \) (outsider) can also register, login and receive his/her smart card.

3. Review of Wu et al.’s Scheme

In this section, we perform an analysis on Wu et al.’s scheme in order to scrutinize the security weakness of their scheme in the next section. Wu et al.’s scheme consists of four phases: registration phase, login phase, authentication phase, and password change phase. In addition, it applies ECC such as the [17] schemes. To begin with, \( \text{GWN} \) creates \( G \) on \( E(F_p) \) with \( P \) as a generator and large
prime $n$ as an order. After that, GWN picks a private key $x$ under two hash functions $h(\cdot)$, $h_1(\cdot)$ and security length $l_s$. In their scheme, they assume that the length of all random numbers should be above $l_s$. Other notations used in Wu et al.'s scheme are abridged in Table 1.

### Table 1. Notations used in this paper.

| Notations | Description |
|-----------|-------------|
| $U_i$     | The $i$-th user |
| $S_i$, $SID_i$ | A $j$-th sensor and its identity |
| $ID_i$    | $U_i$'s identification |
| $PW_i$    | Password of $U_i$ |
| $B_i$     | $U_i$'s Biometric information summarized |
| $A$       | An evil-minded attacker |
| $x$       | Secret key of GWN |
| $r_i$     | Random number generated by $U_i$ |
| $h(\cdot), h_1(\cdot)$ | One-way hash function |
| $\bigoplus$ | Bitwise XOR operator |
| $E(F_p)$  | A group of points on a finite field $F_p$ elliptic curve |
| $P$       | A point generator in $F_p$ with a large prime order $n$ |
| $G$       | A cyclic addition group under $P$ as a generator |
| $sk_{u}, sk_{s}$ | The session key generated by $U_i$ and $S_i$, respectively |

### 3.1. Registration Phase

Registration phase is divided into two parts: user registration phase and registration phase.

#### 3.1.1. User Registration

1. The user $U_i$ first decides his/her identification $ID_i$ and password $PW_i$. With a random number $r_i$, it imprints $B_i$ over a device for biometrics collection, and calculates $Gen(B_i) = (R_i, P_{bi})$, $DID_i = h\left(ID_i || r_i\right)$ and $HPW_i = h\left(PW_i||r_i||R_i\right)$. He/she then requests the registration message $\{ID_i, DID_i\}$ to the gateway node GWN over a secure channel.

2. After the registration request message from the $U_i$ is received, GWN computes $B_i' = h\left(DID_i || x\right)$, where $x$ is GWN's secret key, prepares a smart card for $U_i$ containing $h(\cdot), h_1(\cdot), P$, and collects $ID_i$ in the database. The next thing is that GWN sends the smart card with $B_i'$ to the $U_i$ securely.

3. When receiving the smart card with $B_i'$ from the GWN, $U_i$ computes $B_1 = B_1' \oplus HPW_i$ and $B_2 = h\left(ID_i || R_i || PW_i\right) \oplus r_i$ with storing $B_1, B_2, P$ and $P_{bi}$ in the smart card.

#### 3.1.2. Sensor Registration

1. GWN determines an identity $SID_j$ for new sensor node $S_j$, computes hash function $c_j = h\left(SID_j || x\right)$, and sends $\{SID_j, c_j\}$ to $S_j$.

2. $S_j$ stores $P, SID_j$ and $c_j$, and enters the WSN.

### 3.2. Login Phase

1. $U_i$ enters $ID_i$, $PW_i$ and $B_i'$. Then, the smart card computes $Rep\left(B_i', P_{bi}\right) = R_i, r_i = B_2 \oplus h\left(ID_i || R_i || PW_i\right), HPW_i = h\left(PW_i||r_i||R_i\right)$ and $DID_i = h\left(ID_i || r_i\right)$.

2. The smart card produces random numbers $r_i^{new}, e_i$ and $a \in [1, n-1]$, and selects a special sensor $SID_j$. Then, the smart card calculates $DID_i^{new} = h\left(ID_i || r_i^{new}\right), C_1 = B_1 \oplus HPW_i \oplus e_i, C_2 = aP, \ C_3 = h\left(e_i\right) \oplus DID_i^{new}, Z_i = ID_i \oplus h\left(e_i || DID_i\right)$ and $C_4 = h\left(ID_i || e_i || DID_i || DID_i^{new} || C_2 \ ||\ \ SID_j\right)$. The value $C_4$ is used to certify the integrity of the identities and the new data generated by the user side as well as to authenticate the source of the message $M_1$.

3. $U_i$ sends the login request messages $M_1 = \{C_1, C_2, C_3, Z_i, DID_i, SID_j\}$ to GWN.
3.3. Authentication Phase

1. After the login request messages $M_1$ arrives from the user $U_i$, GWN first computes $e_i = C_1 \oplus h (DID_i \parallel x)$, $DID_i^{new} = C_3 \oplus h (e_i)$ and $ID_i = Z_i \oplus h (e_i \parallel DID_i)$, and verifies the legitimacy of $ID_i$ and $C_4 \overset{?}{=} h (ID_i \parallel e_i \parallel DID_i \parallel DID_i^{new})$. GWN terminates the session if either verification fails. If three failures continuously occur in a certain time span as defined, $U_i$'s account will be frozen; otherwise, GWN calculates $c_j = h (SID_j \parallel x)$ and $C_5 = h (c_j \parallel DID_i \parallel SID_j \parallel C_2)$ and sends $M_2 = \{C_2, C_3, DID_i\}$ to the sensor node $S_j$. The value $C_5$ is used to accredit the integrity of the strings containing $c_j$, and the data can be used for the sensor $S_j$ to acquire the correct data for calculating the session key. This is also done for verification of the source of $M_2$.

2. $S_j$ checks the validity of $C_5$, $C_5 \overset{?}{=} h (c_j \parallel DID_i \parallel SID_j \parallel C_2)$ with its identity $SID_j$. If this step fails, $S_j$ will terminate the session. Otherwise, $S_j$ then chooses $\beta \in [1, n-1]$ and calculates $C_6 = \beta P, sk_b = \beta C_2, C_7 = h_1 (C_2 \parallel C_6 \parallel sk_b \parallel DID_i \parallel SID_j)$ and $C_8 = h (DID_i \parallel SID_j \parallel c_j)$. The main functionality of $C_7$ is used for checking the integrity of the session key and $C_6$, which is needed by $U_i$ to compute the session key. Both $C_7$ and $C_8$ are also used to validate the source of $M_3$. In the end, $S_j$ sends $M_3 = \{C_6, C_7, C_8\}$ to GWN.

3. GWN checks $C_6 \overset{?}{=} h (DID_i \parallel SID_j \parallel c_j)$. If the validation phase fails, GWN terminates the session; otherwise, GWN computes $C_9 = h (DID_i^{new} \parallel x) \oplus h (DID_i \parallel e_i)$ and $C_{10} = h (ID_i \parallel SID_j \parallel DID_i \parallel DID_i^{new} \parallel e_i \parallel C_9)$. The value $C_{10}$ is to check the validation of the source’s message $M_4$. Eventually, GWN sends the message $M_4 = \{C_6, C_7, C_8, C_{10}\}$ to $U_i$.

4. $U_i$ checks $C_{10} \overset{?}{=} h (ID_i \parallel SID_j \parallel DID_i \parallel DID_i^{new} \parallel e_i \parallel C_9)$. $U_i$ then computes the session key $sk_u = \alpha C_6$, and checks $C_7 = h_1 (C_2 \parallel C_6 \parallel sk_b \parallel DID_i \parallel SID_j)$. $U_i$ terminates the session if $U_i$ fails the verification phase. Otherwise, $U_i$ computes $HPW_i^{new} = h (PW_i \parallel r_i^{new} \parallel R_i), B_i^{new} = C_9 \oplus h (DID_i \parallel DID_i^{new} \parallel e_i) \oplus HPW_i^{new}$ and $B_2^{new} = h (ID_i \parallel R_i \parallel PW_i) \oplus r_i^{new}$, and replaces $(B_1, B_2)$ with $(B_1^{new}, B_2^{new})$ in each smart card separately.

3.4. Password and Biometrics Change Phase

1. Same as the step 1 in the Login phase.

2. The smart card produces random numbers $r_i^{new}$ and $e_i$, calculates $DID_i^{new}, C_1, C_3, Z_i$ and $C_{11} = h (ID_i \parallel e_i \parallel DID_i \parallel DID_i^{new})$, and sends $M_5 = \{C_1, C_3, Z_i, C_{11}, DID_i\}$ with a password change request to GWN. The value $C_{11}$ is similar to $C_4$, which is to confirm the integrity of the identities as well as the source of $M_5$.

3. GWN obtains $e_i, ID_i$ and $DID_i^{new}$ as in step 1 of the authentication phase, and checks $ID_i$ and $C_{11} \overset{?}{=} h (ID_i \parallel e_i \parallel DID_i \parallel DID_i^{new})$. If the verification stage fails, GWN terminates the session; otherwise, GWN computes $C_9 = h (DID_i^{new} \parallel x) \oplus h (DID_i \parallel e_i)$ and $C_{12} = h (ID_i \parallel DID_i \parallel DID_i^{new} \parallel e_i \parallel C_9)$ and sends $M_6 = \{C_9, C_{12}\}$ and a grant to $U_i$. Here, $C_{12}$ is to verify the source of $M_6$.

4. $U_i$ checks $C_{12} \overset{?}{=} h (ID_i \parallel DID_i \parallel DID_i^{new} \parallel e_i \parallel C_9)$. If two values are not equal, then $U_i$ terminates this session; otherwise, $U_i$ inputs a new password $PW_i^{new}$ and a new biometric information $B_i^{new}$. The next thing is that the smart card computes $Gen (B_i^{new}) = (R_i^{new}, R_{bi}^{new}), HPW_i^{new2} = h (PW_i^{new} \parallel r_i^{new} \parallel R_i^{new}), B_1^{new2} = C_9 \oplus h (DID_i \parallel e_i) \oplus HPW_i^{new2}$ and $B_2^{new2} = h (ID_i \parallel R_i^{new} \parallel PW_i^{new}) \oplus r_i^{new}$. Finally, $U_i$ substitutes $(B_1^{new2}, B_2^{new2}, R_{bi}^{new2})$ for $(B_1, B_2, P_{bi})$ in the smart card, respectively.

4. Cryptanalysis of Wu et al.'s Scheme

We show that Wu et al.'s scheme [1] possesses certain some security vulnerabilities in this section. The following problems have been found and are described in detail below.
4.1. Extract Critical Information

1. An attacker \( \mathcal{A} \) who is a legitimate user and he/she can own his/her smart card. The smart card can extract the value \{\( B_{1\mathcal{A}}, B_{2\mathcal{A}}, P, P_{b\mathcal{A}} \)\}.

2. \( \mathcal{A} \) can thus obtain \( h(DID_{\mathcal{A}} \parallel x) = B_{1\mathcal{A}} \oplus HPW_{\mathcal{A}} \) and use this variable for other attacks because this value is a critical value that be used on the user identification in the GWN.

4.2. No User Anonymity

Attacker \( \mathcal{A} \) can extract the identity of \( U_i \) from the login request message \( M_i \) of \( U_i \). Assume that \( \mathcal{A} \) eavesdrops on the login request message \( M_1 = \{ C_1, C_2, C_3, C_4, Z_i, DID_i, SID_i \} \) of \( U_i \). We also assume that attacker \( \mathcal{A} \) has \( h(DID_{\mathcal{A}} \parallel x) \) through 5.1. Extract Critical Information. The details are as follows:

1. Attacker \( \mathcal{A} \) first generates random numbers \( r^{new}_{A}, e_A \), and \( a_A \in [1, n - 1] \), and selects a special sensor \( SID_j \). \( C_{1A} = B_{1A} \oplus HPW_A \oplus e_A, C_{2A} = a_A P, C_{3A} = h(e_A) \oplus DID_i, Z_A = ID_A \oplus h(e_A) \oplus DID_A \) and \( C_{4A} = h(ID_A \parallel e_A \parallel DID_A \parallel DID_j \parallel Z_A \parallel SID_A) \).

2. \( \mathcal{A} \) forwards the login request message \( M_{1\mathcal{A}} = \{ C_{1A}, C_{2A}, C_{3A}, C_{4A}, Z_{\mathcal{A}}, DID_{\mathcal{A}}, SID_j \} \) to the gateway node GWN.

3. After receiving the login request message from \( \mathcal{A} \), GWN computes \( e_A = C_{1A} \oplus h(DID_A \parallel x) \), \( DID_1 = C_{3A} \oplus h(e_A) \) and \( ID_A = Z_A \oplus h(e_A) \parallel DID_A \) of \( \mathcal{A} \), and checks the validity of \( ID_A \) and \( C_{4A} = h(ID_A \parallel e_A \parallel DID_A \parallel DID_j \parallel C_{2A} \parallel SID) \). GWN then computes \( c_j = h(SID_j \parallel x) \) and \( C_{5A} = h(c_j \parallel DID_j \parallel SID_j \parallel C_{2A}) \) and sends \( M_{2A} = \{ C_{2A}, C_{5A}, DID_{\mathcal{A}} \} \) to \( S_j \).

4. \( S_j \) checks \( C_{5A} = h(c_j \parallel DID_A \parallel DID_j \parallel SID_j \parallel C_{2A}) \) with its identity \( SID_j \). If this does not hold, \( S_j \) terminates the session. \( S_j \) then selects \( b_A \in [1, n - 1] \) and computes \( C_{6A} = b_A P, sk_s = b_A C_{2A}, C_{7A} = h(C_{6A} \parallel C_{6A} \parallel sk_s \parallel DID_A \parallel SID_j) \) and \( C_{8A} = h(DID_A \parallel SID_j \parallel c_j) \). \( S_j \) sends \( M_{3A} = \{ C_{6A}, C_{7A}, C_{8A} \} \) to GWN.

5. GWN tests \( C_{8A} = h(DID_A \parallel SID_j \parallel c_j) \). If this does not hold, GWN terminates the session; otherwise, GWN calculates \( C_{9A} = h(DID_j \parallel x) \oplus h(DID_A \parallel e_A) \) and \( C_{10A} = h(ID_A \parallel SID_j \parallel DID_A \parallel DID_j \parallel e_A \parallel C_{9A}) \). Finally, GWN sends the message \( M_{4A} = \{ C_{6A}, C_{7A}, C_{9A}, C_{10A} \} \) to attacker \( \mathcal{A} \).

6. \( \mathcal{A} \) calculates \( h(DID_j \parallel x) = h(DID_A \parallel e_A) \oplus C_{9A} \). Now, \( \mathcal{A} \) can compute \( e_i = C_1 \oplus h(DID_i \parallel x) \). Eventually, \( \mathcal{A} \) can find \( ID_i = h(e_i \parallel DID_i) \oplus Z_i \).

This result shows that Wu et al.’s scheme does not ensure user anonymity.

4.3. User Impersonation Attack

An attacker \( \mathcal{A} \) can impersonate any user through the identity of others and his/her own information. We assume the casualty is \( U_i \). We also assume that attacker \( \mathcal{A} \) has \( h(DID_A \parallel x) \) through Section 5.1. Extract Critical Information. The detailed method is as follows:

1. Attacker \( \mathcal{A} \) selects \( ID_i \) who is the target of the user impersonation attack.

2. \( \mathcal{A} \) selects random numbers \( r^{new}_{A}, e_A \), and \( a_A \in [1, n - 1] \) and selects a particular sensor \( SID_j \). Then, \( \mathcal{A} \) calculates \( DID^{new}_{A} = h(ID_A \parallel r^{new}_{A}), C_{1A} = B_{1A} \oplus HPW_A \oplus e_A, C_{2A} = a_A P, C_{3A} = h(e_A) \oplus DID_A, Z_A = ID_A \oplus h(e_A) \oplus DID_A \) and \( C_{4A} = h(ID_A \parallel e_A \parallel DID_A \parallel DID^{new}_A \parallel C_{2A} \parallel SID) \). \( C_{4A} \) is to check the new data produced on the user side and the integrity of the identities as well as to verify the source of \( M_{1A} \).

3. \( \mathcal{A} \) forwards the login request message \( M_{1A} = \{ C_{1A}, C_{2A}, C_{3A}, C_{4A}, Z_{\mathcal{A}}, DID_{\mathcal{A}}, SID_j \} \) to GWN.

4. After obtaining the message from the \( \mathcal{A} \), GWN calculates \( e_A = C_{1A} \oplus h(DID_A \parallel x), DID^{new}_{A} = C_{3A} \oplus h(e_A) \) and \( ID_1 = Z_A \oplus h(e_A \parallel DID_A) \), and checks the availability of \( ID_1 \) and checks \( C_{4A} = h(ID_1 \parallel e_A \parallel DID_{\mathcal{A}} \parallel DID^{new}_{A} \parallel C_{2A} \parallel SID_j) \). GWN continues to proceed with the scheme without detection. Unfortunately, the GWN mistakenly believes that he/she is communicating with the legitimate patient \( U_i \).
Resultingly, the attacker $A$ will be successfully confirmed as $GWN$ by user $U_i$. Hence, the user impersonation attack is successful.

In the next section, we discuss Wu et al.’s scheme to overcome the weakness of the scheme. Our scheme stores several variables in the database to prevent the vulnerability of Wu et al.

5. Proposed Scheme

We propose a new three-factor user authentication scheme for wireless sensor networks in this section. We use three participants: the user $U_i$, the gateway node $GWN$ and the sensor node $S_j$. The gateway node $GWN$ creates master keys $x$. The user $U_i$ and the sensor node $S_j$ computes on elliptic curve group $F_p$.

We have defined the name of the variable as follows:

- $G_1, G_2, G_3$: Generator of smart card,
- $MU_1, MU_2, MU_3$: message sent by user,
- $MG_1, MG_2, MG_3, MG_4$: message sent by gateway node,
- $MS_1, MS_2, MS_3$: message sent by the server node.

Other variables do not have that special meaning. The proposed scheme is composed as follows: registration phase, login phase, authentication phase, and password/biometrics change phase.

5.1. Registration Phase

In this phase, a user $U_i$ chooses an identity $ID_i$, imprints biometric template $B_i$ at the sensor, and then performs the following steps:

5.1.1. User Registration Phase

1. $U_i$ selects $ID_i$ and $PW_i$. imprints $B_i$ via a device for biometrics collection and computes $Gen(B_i) = (R_i, P_{bi})$ and $HPW_i = h(ID_i \parallel PW_i \parallel R_i)$. Then, he/she sends $ID_i$ to $GWN$ secretly.
2. $GWN$ generates a random number $r_i$ and computes $GID_i = h(ID_i \parallel r_i)$.
3. $GWN$ computes $G'_i = h(GID_i \parallel x)$, prepares a smart card for $U_i$ containing $h(\cdot)$, $h_1(\cdot)$, $P$, $GID_i$ and the fuzzy extractor.
4. $GWN$ stores $ID_i$ and $GID_i$ in its database and shares it with $U_i$. By storing $ID_i$ and $GID_i$ in the database, Wu et al. [1]’s problems arising from existing $DID_i$ can be solved.
5. $U_i$ computes $G_1 = G'_i \oplus HPW_i$, $G_2 = h(ID_i \parallel R_i \parallel PW_i) \oplus GID_i$ and $G_3 = h(ID_i \parallel GID_i)$. $\{G_1, G_2, G_3, h(\cdot), h_1(\cdot), P\}$ are stored in the smart card.

5.1.2. Sensor Registration Phase

1. $GWN$ selects an identity $SID_j$ for each new sensor $S_j$, computes $c_j = h(SID_j \parallel x)$ and sends $\{SID_j, c_j\}$ to $S_j$.
2. $S_j$ stores $P$, $SID_j$ and $c_j$ and joins the WSN.

Figure 1 illustrates the registration phase of the proposed scheme.
Figure 1. Registration phase of the proposed scheme.

5.2. Login Phase

1. $U_i$ inputs $ID_i$, $PW_i$ and $B'_i$. The smart card executes $Rep(B'_i, P_{bi}) = R_i$ and $GID_i = G_2 \oplus h(ID_i || R_i || PW_i)$. $U_i$ checks $h(ID_i || GID_i) \neq G_3$. This allows $U_i$ to verify whether it has come in correctly.

2. $U_i$ generates $e_i$ and $a$. $U_i$ computes $HPW_i = h(ID_i || PW_i || R_i)$, $MU_1 = G_1 \oplus HPW_i \oplus e_i$, $MU_2 = aP$ and $MU_3 = h(ID_i || e_i || GID_i || MU_2 || SID_j)$. $U_i$ sends the message $M_1 = \{MU_1, MU_2, MU_3, GID_i, SID_j\}$ to GWN.

Figure 2 illustrates the login and authentication phase of the proposed scheme.

5.3. Authentication Phase

1. $GWN$ finds $ID_i$ by using $GID_i$ from the database and computes $e_i = MU_1 \oplus h(GID_i || x)$. $GWN$ checks the validity of $MU_3 \neq h(ID_i || e_i || GID_i || MU_2 || SID_j)$. If it fails, the session will be terminated. Otherwise, $GWN$ computes $c_j = h(SID_j || x)$ and $MG_1 = h(c_j || GID_i || SID_j || MU_2)$. When the operation has finished, $GWN$ sends the message $M_2 = \{MU_2, MG_1, GID_i\}$ to $S_j$.

2. $S_j$ checks $MG_1 \neq h(c_j || GID_i || SID_j || MU_2)$ with its identity $SID_j$. If it is wrong, $S_j$ will stop the session. Otherwise, $S_j$ selects $\beta \in [1, n - 1]$ and computes $MS_1 = \beta P$, session key $sk_s = \beta MU_2$, $MS_2 = h_1(MU_2 || MS_1 || sk_s || GID_i || SID_j)$ and $MS_3 = h(GID_i || SID_j || c_j)$. It sends message $M_3 = \{MS_1, MS_2, MS_3\}$ when all operations have finished.

3. $GWN$ checks $MS_3 \neq h(GID_i || SID_j || c_j)$. If it is wrong, the session will be stopped. Otherwise, $GWN$ generates $r_{new}^i$ and calculates $GID_{new}^i = h(ID_i || r_{new}^i)$, $MG_2 = h(GID_{new}^i || x) \oplus h(GID_i || e_i)$, $MG_3 = h(ID_i || SID_j || GID_i || GID_{new}^i || e_i || MG_2)$ and $MG_4 = h(e_i) \oplus GID_{new}^i$. Finally, $GWN$ sends the message $M_4 = \{MS_1, MS_2, MG_2, MG_3, MG_4\}$ to $U_i$.

4. $U_i$ computes $GID_{new}^i$ and checks $MG_3 \neq h(ID_i || SID_j || GID_i || GID_{new}^i || e_i || MG_2)$. If not, the session will be stopped. $U_i$ computes $sk_u = \alpha MS_1 = \alpha \beta P$ and checks $MS_2 \neq h_1(MU_2 || MS_1 || sk_u || GID_i || SID_j)$. If it is wrong, $U_i$ will stop the session.
5. $U_i$ computes $G_1^{new} = MG_2 \oplus h(GID_i \parallel e_i) \oplus HPW_i$, $G_2^{new} = G_2 \oplus GID_i \oplus GID^{new}$ and $G_3^{new} = h(ID_i \parallel GID^{new})$. Finally, $U_i$ substitutes ($G_1^{new}, G_2^{new}, G_3^{new}$) for ($G_1, G_2, G_3)$ in the smart card, respectively.

- User $U_i$
  - (ID$_i$, PW$_i$, smart card)
- Inserts smart card
- Inputs ID$_i$, PW$_i$, and biometric B'_i
- $R'_i = \text{Rep}(B'_i, R_i)$. GID$_i = G_2 \oplus h(ID_i \parallel |PW_i$)
- $h(ID_i \parallel GID)_i \triangleq G_2$
- Generate $e_i$, $\alpha$
- $HPW_i = h(ID_i \parallel |PW_i) \parallel R_i$
- $MU_i = G_1 \oplus HPW_i \oplus e_i$
- $MU_2 = \alpha \beta$
- $MU_3 = h(ID_i \parallel |e_i) \parallel GID_i \parallel MU_2 || SID_i)$

\[ (MU_1, MU_2, MU_3, GID_i, SID_i) \]

\[ MU_2 \triangleq h(ID_i \parallel |e_i) \parallel GID_i \parallel MU_2 || SID_i \]
\[ c_j = h(SID_j || x) \]
\[ MG_1 = h(c_j || GID_i || MU_2) \]
\[ MG_1 \triangleq h(c_j || GID_i || MU_2) \]
\[ Generate \ \beta \]
\[ MS_1 = \beta \]
\[ sk_1 = \beta MU_2 = \alpha \beta \]
\[ MS_2 = h(MU_2 || MS_1 || sk_1 || GID_i || SID_i) \]
\[ MS_3 = h(GID_i || SID_i) \]

\[ (MS_1, MS_2, MS_3) \]

\[ GID^{new} = MG_2 \parallel h(GID^{new} || e_i) \]
\[ MG_2 = h(GID^{new} || x) \parallel h(GID^{new} || e_i) \]
\[ MG_3 = h(ID_i || SID_i \parallel GID_i || GID^{new} || e_i || MG_2) \]
\[ MG_4 = h(e_i) \parallel GID^{new} \]
\[ GID^{new} \text{ stores in database} \]

\[ (MS_1, MS_2, MG_2, MG_3, MG_4) \]

\[ Shared \ sk = \alpha \beta \]

**Figure 2.** Login and authentication phase of the proposed scheme.

### 5.4. Password and Biometrics Change Phase

1. $U_i$ inputs ID$_i$, PW$_i$, and B'$_i$. The smart card executes $\text{Rep}(B'_i, P_b) = R_i$ and $GID_i = G_2 \oplus h(ID_i \parallel R_i \parallel PW_i)$. $U_i$ checks $h(ID_i \parallel GID_i) \triangleq G_3$. This allows $U_i$ to verify whether it has come in correctly.

2. $U_i$ is asked to input a new password PW'$_{i,new}$ and new biometric information B'$_ {i,new}$. The following data are computed: $\text{Gen}(G_1^{new} = (R_i^{new}, P_{b_i}^{new}), HPW_i^{new} = h(ID_i \parallel PW_i^{new} || R_i^{new}), G_2^{new} =$
\[ G_1 \oplus HPW_i \oplus HPW_i^{new}, G_2^{new} = G_2 \oplus h (ID_i \parallel R_i \parallel PW_i) \oplus h (ID_i \parallel R_i \parallel PW_i^{new}). \] Finally, \( U_i \) substitutes \((G_1^{new}, G_2^{new}, P_{bi})\) for \((G_1, G_2, P_{bi})\) in the smart card, respectively.

6. Security Analysis of the Proposed Scheme

6.1. Formal Security Analysis

The formal security analysis uses an automated analysis tool called ProVerif. ProVerif is an automated tool for analyzing cryptographic protocols that was developed by Bruno Blanchet. Digital signatures, hash functions, signature proofs, etc. are suitable for analyzing an authentication protocol. Recently, many researchers \([1,4,24]\) have verified the authentication in the user authentication protocol using ProVerif. The formal security analysis shows the results of verifying and analyzing the security of the proposed scheme using ProVerif.

We use three channels. We provide the illustration of Table 2. \( cha \) is the channel in the registration phase and is used when the user \( U_i \) and \( GWN \) exchange \( ID_i \) in the registration phase. \( chc \) is the channel used by user \( U_i \) and \( GWN \) to exchange messages in the login phase and \( chb \) is used when the \( GWN \) and Sensor node \( S_j \) exchange messages in the login phase. Five initial variables were used: \( R_i, ID_i, ID_g, SID_j, \) and \( PW_i, ID_i, \) and \( PW_i \) are the personal information made by the user \( U_i \) when registering. \( R_i \) is a random string made up of the user’s biometric information. \( ID_g \) is the identity of the gateway and \( SID_j \) is the unique string of the sensor node \( S_j \). \( x \) is defined as a secret key. \( P \) is a generator for creating a session key, which is the initial value used in ECC. The concatenate function and the \( xor \) function, including the multiplication in ECC and the hash function \( h \) and \( h_1 \), are defined for the events that indicate the start and end of each.

| Table 2. Define values and functions. |
|--------------------------------------|
| (*—channels—*)                       |
| free cha:channel [private].           |
| free chb:channel.                    |
| free chc:channel.                    |
| (*—constants—*)                      |
| free Ri:bitstring [private].          |
| free IDi:bitstring [private].         |
| free IDg:bitstring.                  |
| free SIDj:bitstring.                 |
| free PWi:bitstring [private].         |
| (*—secret key—*)                     |
| free x:bitstring [private].           |
| (*—shared key—*)                     |
| free P:bitstring [private].           |
| (*—functions—*)                      |
| fun concat(bitstring, bitstring):bitstring. |
| fun xor(bitstring, bitstring):bitstring. |
| fun h(bitstring):bitstring.           |
| fun h1(bitstring):bitstring.          |
| fun mult(bitstring, bitstring):bitstring. |
| equation forall a:bitstring, b:bitstring; mult(a, b) = mult(b, a). |
| equation forall a:bitstring, b:bitstring; xor(xor(a, b), b) = a. |
| (*—events—*)                         |
| event beginUi(bitstring).             |
| event endUi(bitstring).               |
| event beginGWN(bitstring).            |
| event endGWN(bitstring).              |
| event beginSj(bitstring).             |
| event endSj(bitstring).               |
Table 3 shows the registration phase of the user $U_i$ and the process of the login and authentication phase. Table 4 demonstrates the registration phase and the login and authentication phase of the GWN. Table 5 displays the authentication phase of the sensor node $S_j$. Table 6 shows the query against the attack with the prover-sive, and Table 7 shows the result for Table 6.

**Table 3. $U_i$ protocol.**

(*—-$U_i$ process—*)

let $U_i =$

let $H_{PW_i} = h(\text{concat}(\text{concat}(I_{Di}, P_{Wi}), R_{i}))$ in

out(cha, $(I_{Di})$);

in(cha, $(X_{GID_i}$:bitstring));

let $G_1' = h(\text{concat}(X_{GID_i}, x))$ in

let $G_1 = \text{xor}(G_1', H_{PW_i})$ in

let $G_2 = xor(h(\text{concat}(\text{concat}(I_{Di}, R_{i}), P_{Wi})), X_{GID_i})$ in

let $G_3 = h(\text{concat}(I_{Di}, X_{GID_i})$ in

event beginUi(IDi);

new ei:bitstring;

new alpha:bitstring;

let $GID_i = \text{xor}(G_2, h(\text{concat}(\text{concat}(I_{Di}, R_{i}), P_{Wi})))$ in

if $h(\text{concat}(I_{Di}, X_{GID_i})) = G_3$ then

let $H_{PW_i} = h(\text{concat}(\text{concat}(I_{Di}, P_{Wi}), R_{i}))$ in

let $MU_1 = \text{xor}(xor(G_1, H_{PW_i}), ei)$ in

let $MU_2 = \text{mult}(\text{alpha}, P)$ in

let $MU_3 = h(\text{concat}(\text{concat}(I_{Di}, ei), \text{concat}(\text{concat}(X_{GID_i}, MU_2), SID_j)))$ in

out(chc, $(MU_1, MU_2, MU_3, GID_i, SID_j)$);

in(chc, $(XXMS_1$:bitstring, $XXMS_2$:bitstring, $XXMS_3$:bitstring, $XXMS_4$:bitstring));

let $GID_{inew} = \text{xor}(XXMS_4, h(ei))$ in

if $XXMS_3 = h(\text{concat}(\text{concat}(I_{Di}, SID_j), \text{concat}(\text{concat}(GID_i, GID_{inew}), \text{concat}(ei, XXMS_2))))$ then

let $sku = \text{mult}(\text{alpha}, XXMS_1)$ in

let $GID_{inew} = h(\text{concat}(XXMS_1, ri))$ in

let $G_1' = h(\text{concat}(GID_i, x))$ in

out(cha, $(GID_i)$);

in(chc, $(X_{GID_i}$:bitstring, $X_{GID_i2}$:bitstring, $XGID_i3$:bitstring, $XGID_i4$:bitstring));

event beginGWN(IDg);

let $ei = \text{xor}(XXMS_1, h(\text{concat}(XXMS_1, x)))$ in

if $XXMS_3 = h(\text{concat}(\text{concat}(X_{GID_i}, ei), \text{concat}(\text{concat}(XXMS_2, XXMS_3), \text{concat}(XXMS_4, SID_j))))$ then

let $cj = h(\text{concat}(XXMS_4, x))$ in

let $MG_1 = h(\text{concat}(\text{concat}(X_{GID_i}, X_{GID_i2}), \text{concat}(XXMS_3, XXMS_4)))$ in

out(chb, $(XXMS_1, XXMS_2)$);

in(chb, $(XXMS_1$:bitstring, $XXMS_2$:bitstring, $XXMS_3$:bitstring, $XXMS_4$:bitstring));

if $XXMS_3 = h(\text{concat}(\text{concat}(X_{GID_i}, SID_j), cj))$ then

new $rinew$:bitstring;

let $GID_{inew} = h(\text{concat}(XXMS_1, rinew))$ in

let $MG_2 = xor(h(\text{concat}(GID_{inew}, x)), h(\text{concat}(XXMS_2, ei)))$ in

let $MG_3 = h(\text{concat}(\text{concat}(X_{GID_i}, SID_j), \text{concat}(\text{concat}(XXMS_3, GID_{inew}), \text{concat}(ei, MG_2))))$ in

let $MG_4 = xor(h(ei), GID_{inew})$ in

out(chc, $(XXMS_1, XXMS_2, MG_2, MG_3, MG_4)$);

event endGWN(IDg).
Table 5. $S_j$ protocol.

(*—$S_j$ process—*)

```plaintext
let $S_j =$
in(chb, (XXMU2:bitstring, XMG1:bitstring, XXGIDi:bitstring));
event begin$S_j$(SIDj);
let scj = $h$(concat(SIDj, x)) in
if XMG1 = $h$(concat(concat(scj, XXGIDi), concat(SIDj, XXMU2))) then
new beta:bitstring;
let MS1 = mult(beta, P) in
let sks = mult(beta, XXMU2) in
let MS2 = $h1$(concat(concat(XXMU2, MS1), concat(concat(sks, XXGIDi), SIDj))) in
let MS3 = $h$(concat(concat(XXGIDi, SIDj), scj)) in
out(chb, (MS1, MS2, MS3));
event end$S_j$(SIDj).
```

Table 6. Queries.

(*—queries—*)

```plaintext
query attacker(P).
query id:bitstring; inj-event(endUi(id)) ==> inj-event(beginUi(id)).
query id:bitstring; inj-event(endGWN(id)) ==> inj-event(beginGWN(id)).
query id:bitstring; inj-event(end$S_j$(id)) ==> inj-event(begin$S_j$(id)).
```

Table 7. Output of queries.

RESULT inj-event(end$S_j$(id)) ==> inj-event(begin$S_j$(id)) is true.
RESULT inj-event(endGWN(id_12209)) ==> inj-event(beginGWN(id_12209)) is true.
RESULT inj-event(endUi(id_25655)) ==> inj-event(beginUi(id_25655)) is true.
RESULT not attacker(P[]) is true.

When the code that makes up the scheme is executed, ProVerif prints the following results:

1. RESULT inj-event(EVENT) ==> inj-event(EVENT) is true.
2. RESULT inj-event(EVENT) ==> inj-event(EVENT) is false.
3. RESULT (QUERY) is true.
4. RESULT (QUERY) is false.

The first code means that the event has been verified and the authentication has been successful, while the second code means that the event has not been verified. The third code means that the query was proven and the attack was not successful. When the fourth code is displayed, the query is false, meaning that an attack is possible and the attack induction and tracking is thus displayed.

The ProVerif result of the proposed scheme is shown to be accurate for all events by simulating the result as shown in the figure (see Table 8). Therefore, the proposed scheme is safe from virtual attacker A and the virtual attack has been successfully terminated.
**Table 8. Performance comparison.**

| Features                                 | Wu et al. [1] | Park et al. [3] | Park et al. [25] | Ours |
|------------------------------------------|---------------|-----------------|------------------|------|
| Defence of privileged insider attack     | O             | O               | O                | O    |
| Defence of outsider attack               | X             | X               | X                | O    |
| Defence of offline ID guessing attack    | O             | O               | O                | O    |
| Defence of online ID guessing attack     | X             | X               | X                | O    |
| Defence of session key disclosure attack | O             | O               | O                | O    |
| Defence of user impersonation attack     | X             | X               | O                | O    |
| Defence of server impersonation attack   | O             | X               | O                | O    |
| User anonymity                           | X             | O               | X                | O    |
| Forward secrecy and backward secrecy     | O             | O               | O                | O    |

### 6.2. Informal Security Analysis

#### 6.2.1. Privileged Insider Attack

The only value that the user sends in the registration center is the \( ID_i \). However, their \( ID_i \) is used after hashing with other values at every subsequent step. It can not be used because it is used as hashed with values that are not exposed to the outside such as \( PW_i \) or \( R_i, GID_i, GID_{new}, e_i, MU_2 \) and \( SID_j, MG_2 \), and these values are not exposed. Therefore, it is safe from a privileged insider attack.

#### 6.2.2. Outsider Attack

\( U_i \)'s smart cards include \( h(\cdot), h_1(\cdot), P, GID_i \), and fuzzy extractors. Information such as session key or \( ID_i \), which can be a critical value, or information such as a user’s password are all hashed, or can not be extracted because the value can not be extracted from ECC. In addition, \( IDs \) and \( GIDs \) are kept in the database, and \( ID_i \) information can not be extracted because \( ID_i \) are not used directly in the protocol.

#### 6.2.3. Offline ID Guessing Attack

\( PW_i \) and \( ID_i \) are not used directly in this phase. They are used through hashing by concatenating them with other variables, so \( ID_i \) and \( PW_i \) can not be directly obtained from public information. Therefore, \( ID_i \) and \( PW_i \) can not be obtained using login request messages \( MU_1, MU_2, MU_3, GID_i \), and \( SID_j \). Since \( ID_i \) and \( GID_i \) are combined and stored in the database, it is impossible to extract the \( ID_i \) from the protocol.

#### 6.2.4. Online ID Guessing Attack

\( ID_i \) and \( PW_i \) are not directly used in the phase so the attacker can not guess the \( ID_i \)s or passwords of others. It is impossible to retrieve a user’s \( ID_i \) in the protocol because the \( IDs \) and \( GIDs \) are stored in the database, and \( ID_i \) is found by searching the database.

#### 6.2.5. Session Key Disclosure Attack

The session key should be computed as \( \beta \) or \( \alpha \) when knowing \( aP \) or \( \beta P \) with \( a\beta P \). Neither \( \beta \) nor \( \alpha \) are known to the user or the sensor node, so it is impossible to know the session key unless it is a user or a sensor node.

#### 6.2.6. User Impersonation Attack

After the \( ID_i \) is found in the database using the \( GID, e_i = MU_1 + h \left( GID_i || x \right) \) is calculated in order to compare the \( MU_3 \) and \( h \left( ID_i || e_i || GID_i || MU_2 || SID_j \right) \). One can never be accepted as a specific user without knowing the \( ID \) and \( GID \) pair. Therefore, a User Impersonation Attack is impossible.
6.2.7. Server Impersonation Attack

The server is identified in $MS_3 = h(GID_i||SID_j||c_j)$. $c_j = h(SID_j||x)$ and $x$ is the secret key. Therefore, it is necessary to know the $c_j$ calculated by the secret key other than the $GID_i$ and the $SID_j$ included in the message in order to authenticate the server and $c_j$ is not used alone and $MG_1 = h(c_j||GID_i||SID_j||MU_2)$, $MS_3 = h(GID_i||SID_j||c_j)$ and other values. In addition, the value $x$ in the destination $c_j = h(SID_j||x)$ can not be determined because it is always used by hashing with $SID_j$.

6.2.8. User Anonymity

In the login process, the user gives $MU_1$, $MU_2$, $MU_3$, $GID_i$, and $SID_j$ to the GWN. In this case, $GID_i = G_2 + h(ID_i||R_i||PW_i)$ is continuously changed by the random number $R_i$. Since $ID_i$ is used by hashing, one cannot guess $ID_i$ through $MU_1$, $MU_2$, $MU_3$, $GID_i$, and $SID_j$.

6.2.9. Forward Secrecy and Backward Secrecy

Because of the nature of ECCDH, we can not find $\alpha P$ and $\beta P$ through $\alpha \beta P$, we can not find $\alpha \beta P$ through $\alpha P$ and $\beta P$, and we can not find $\alpha$ through $P$ and $\alpha P$.

7. Performance Analysis of the Proposed Scheme

Four symbols in total are used to analyze performance. $T_m$ is the time of the multiplicative operation used in ECC. This takes the most time in our scheme. $T_{Rep}$ assumes that it is equal to $T_m$, the time to check for a match when recognizing the user’s biometric $B_i^*$. $T_s$ means time in symmetric encryption or decryption. Finally, $T_h$ means the time it takes to use the hash function. These are listed in Table 9.

| Symbol | Meaning | Time (ms) |
|--------|---------|-----------|
| $T_m$  | time of multiplication in Field | 7.3529 [26] |
| $T_{Rep}$ | time of $Rep$ | $= T_m$ [16] |
| $T_s$  | time of symmetric encryption or decryption | 0.1303 [26] |
| $T_h$  | time of hash operation | 0.0004 [26] |

The authors [26] measured the approximate execution time of each cryptographic operation under the following conditions:

- CPU: Intel(R) Core(TM)2T6570 2.1 GHz,
- Memory: 4 G,
- OS: Win7 32-bit,
- Software: Visual C++ 2008,
- MIRACL C/C++ Library,
- Security level: 160-bit point in $F_p$,
- 1024-bit in a cyclic group, AES and SHA-1.

The proposed scheme produced the best results in time among all the three factor user authentication schemes using ECC (see Table 10).
Table 10. Performance comparison.

|                         | Wu et al. [1]                  | Park et al. [3]                  | Park et al. [25]                  | Ours                  |
|-------------------------|--------------------------------|---------------------------------|-----------------------------------|-----------------------|
| User $U_i$ (ms)          | $10T_h + 1T_{Rep} + 2T_m$ = 22.0627 | $6T_h + 1T_{Rep} + 2T_m$ = 22.0611 | $10T_h + 1T_{Rep} + 2T_m$ = 22.0627 | $8T_h + 1T_{Rep} + 2T_m$ = 22.0619 |
| GWN (ms)                | $10T_h$ = 0.004                 | $7T_h + 2T_e$ = 0.2634           | $11T_h$ = 0.0044                  | $10T_h$ = 0.004       |
| Sensor node $S_j$ (ms)   | $2T_h + 2T_m$ = 14.7066        | $6T_h + 2T_m + 3T_e$ = 14.8385   | $4T_h + 2T_m$ = 14.7074           | $3T_h + 2T_m$ = 14.707 |
| Total costs (ms)         | $22T_h + 4T_m + 1T_{Rep}$ = 36.7733 | $19T_h + 4T_m + 3T_e + 1T_{Rep}$ = 37.163 | $25T_h + 4T_m + 1T_{Rep}$ = 36.7745 | $21T_h + 4T_m + 1T_{Rep}$ = 36.7729 |

8. Conclusions

Many user authentication schemes have been proposed for wireless sensor networks, but they have serious security flaws, respectively. Recently, Wu et al. also proposed a three-factor user authentication scheme, which is looking promising. However, we discovered vulnerabilities in the configuration of their scheme and proposed a new scheme to address the discovered issues. Finally, we provide security and performance analysis between the Wu et al. scheme and our proposed protocol, and provide formal analysis based on the ProVerif. The security and performance of the proposed scheme are significantly better than the existing user authentication schemes. Our scheme is not very fast yet. In the future, we will study the WSN protocol, which is safer, simpler and faster.

Author Contributions: J.R., H.L. conceived and designed the experiments; J.R., H.K. performed the experiments; J.R. and D.W. analyzed the data; and J.R. and H.K. wrote the paper.

Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2010-0020210).

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

Appendix A

Table A1. Explanation of each abbreviation.

| Notations | Description |
|-----------|-------------|
| WSN       | Wireless sensor network |
| RSA       | A public-key encryption technology developed by Ron Rivest, Adi Shamir, and Leonard Adleman |
| ECC       | Elliptic curve cryptosystem created by Victor S. Miller and Neal Koblitz |
| Gen       | A probabilistic generation function for which the biometrics $B$ returns a string $\alpha$ and a string $\beta$ |
| $Rep$     | A function that restore $\beta$ to $\alpha$ and any vector $B^*$ close to $B$ |
| $B$       | A vector with biometric information |
| $B^*$     | Any vector $B^*$ close to $B$ |
| GWN       | Gateway node |

References

1. 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]
2. Das, M. Two-Factor User Authentication in Wireless Sensor Networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1086–1090. [CrossRef]
3. Park, Y.; Lee, S.; Kim, C.; Park, Y. Secure biometric-based authentication scheme with smart card revocation/reissue for wireless sensor networks. *Int. J. Distrib. Sens. Netw.* **2016**, *12*, 1–11. [CrossRef]
4. Kumari, S.; Chaudhry, S.A.; Wu, F.; Li, X.; Farash, M.S.; Khan, M.K. An improved smart card based authentication scheme for session initiation protocol. *Peer-to-Peer Netw. Appl.* **2017**, *10*, 92–105. [CrossRef]
5. Elhoseny, M.; Yuan, X.; El-Minir, H.K.; Riad, A.M. An energy efficient encryption method for secure dynamic WSN. Secur. Commun. Netw. 2016, 9, 2024–2031. [CrossRef]
6. Gaber, T.; Abdelwahab, S.; Elhoseny, M.; Hassanien, A.E. Trust-based secure clustering in WSN-based intelligent transportation systems. Comput. Netw. 2018, 146, 151–158. [CrossRef]
7. Watro, R.; Kong, D.; Cuti, S.; Gardiner, C.; Lynn, C.; Kruus, P. TinyPK: Securing Sensor Networks with Public Key Technology. In Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks, Washington, DC, USA, 25–29 October 2004; pp. 59–64.
8. Moon, J.; Choi, Y.; Jung, J.; Won, D. An Improvement of Robust Biometrics-based Authentication and Key Agreement Scheme for Multi-Server Environments using Smart Cards. PLoS ONE 2015, 10, e0126323. [CrossRef] [PubMed]
9. Choo, K.K.R.; Nam, J.; Won, D. A mechanical approach to derive identity-based protocols from Diffie–Hellman-based protocols. Inf. Sci. 2014, 281, 182–200. [CrossRef]
10. Kumar, P.; Lee, H.J. Cryptanalysis on two user authentication protocols using smart card for wireless sensor networks. In Proceedings of the 2011 Wireless Advanced, London, UK, 20–22 June 2011; pp. 241–245.
11. Yeh, H.L.; Chen, T.H.; Liu, P.C.; Kim, T.H.; Wei, H.W. A secured authentication protocol for wireless sensor networks using elliptic curves cryptography. Sensors 2011, 11, 4767–4779. [CrossRef] [PubMed]
12. Jiang, Q.; Ma, J.; Lu, X.; Tian, Y. An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks. Peer-to-Peer Netw. Appl. 2015, 8, 1070–1081. [CrossRef]
13. Das, A.K. A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks. Peer-to-Peer Netw. Appl. 2016, 9, 223–244. [CrossRef]
14. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H. Examining smart-card security under the threat of power analysis attacks. IEEE Trans. Comput. 2002, 51, 541–552. [CrossRef]
15. Lee, H.; Lee, D.; Moon, J.; Jung, J.; Kang, D.; Kim, H.; Won, D. An improved anonymous authentication scheme for roaming in ubiquitous networks. PLoS ONE 2018, 13, e0193366. [CrossRef] [PubMed]
16. Xu, L.; Wu, F. Cryptanalysis and improvement of a user authentication scheme preserving uniqueness and anonymity for connected health care. J. Med. Syst. 2015, 39, 10. [CrossRef] [PubMed]