sensors

MDPI

Article

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

check for updates

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:

$$y^2 = x^3 + ax + b \quad mod\ p \qquad a, b \in F_p. \tag{1}$$

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

$$4a^3 + 27b^2 \neq 0 \quad mod\ p. \tag{2}$$

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 (Gen, Rep), demonstrated as [17,21]:

$$Gen\ (B) \rightarrow \langle \alpha, \beta \rangle\ , \tag{3}$$

$$Rep\ (B^*, \beta) = \alpha. \tag{4}$$

Gen is a probabilistic generation function for which the biometrics B returns a factored out string $\alpha \in \{0,1\}^k$ and a coadjutant string $\beta \in \{0,1\}^*$, and Rep is a function that restores β to α, 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 $\mathcal{A}$ can be a user, a gateway, or a sensor. Any registered user can act as an attacker.
2. $\mathcal{A}$ 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. $\mathcal{A}$ 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 $\mathcal{A}$ (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, 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_j, SID_j	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
$\mathcal{A}$	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
$X\|\|Y$	Concatenation operator
$\oplus$	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_j, 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 $Gen(B_i) = (R_i, P_{bi})$, $DID_i = h(ID_i \parallel r_i)$ and $HPW_i = h(PW_i \parallel r_i \parallel R_i)$. 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 $B_1' = h(DID_i \parallel x)$, 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_1' to the U_i securely.
3. When receiving the smart card with B_1' from the GWN, U_i computes $B_1 = B_1' \oplus HPW_i$ and $B_2 = h(ID_i \parallel R_i \parallel PW_i) \oplus r_i$ with storing B_1, B_2, P and P_{bi} in the smart card.

3.1.2. Sensor Registration

1. GWN determines an identity $SIDj$ for new sensor node S_j, computes hash function $c_j = h(SID_j \parallel x)$, and sends $\{SID_j, c_j\}$ to S_j.
2. S_j stores P, SID_j and c_j, and enters the WSN.

3.2. Login Phase

1. U_i enters ID_i, PW_i and B_i'. Then, the smart card computes $Rep(B_i', P_{bi}) = R_i$, $r_i = B_2 \oplus h(ID_i \parallel R_i \parallel PW_i)$, $HPW_i = h(PW_i \parallel r_i \parallel R_i)$ and $DID_i = h(ID_i \parallel r_i)$.
2. The smart card produces random numbers r_i^{new}, e_i and $\alpha \in [1, n-1]$, and selects a special sensor SID_j. Then, the smart card calculates $DID_i^{new} = h(ID_i \parallel r_i^{new})$, $C_1 = B_1 \oplus HPW_i \oplus e_i$, $C_2 = \alpha P$, $C_3 = h(e_i) \oplus DID_i^{new}$, $Z_i = ID_i \oplus h(e_i \parallel DID_i)$ and $C_4 = h(ID_i \parallel e_i \parallel DID_i \parallel DID_i^{new} \parallel C_2 \parallel SID_j)$. 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 $M_1 = \{C_1, C_2, C_3, C_4, Z_i, DID_i, SID_j\}$ 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 $C_4 \stackrel{?}{=} h(ID_i \parallel e_i \parallel DID_i \parallel DID_i^{new} \parallel C_2 \parallel SID_j)$. 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 $C_5 = h(c_j \parallel DID_j \parallel SID_j \parallel C_2)$ and sends $M_2 = \{C_2, C_5, DID_i\}$ 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, $C_5 \stackrel{?}{=} h(c_j \parallel DID_i \parallel SID_j \parallel C_2)$ 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 $C_6 = \beta P$, $sk_s = \beta C_2$, $C_7 = h_1(C_2 \parallel C_6 \parallel sk_s \parallel DID_i \parallel SID_j)$ 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 $C_8 \stackrel{?}{=} h(DID_i \parallel SID_j \parallel c_j)$. 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 $M_4 = \{C_6, C_7, C_9, C_{10}\}$ to U_i.
4. U_i checks $C_{10} \stackrel{?}{=} h(ID_i \parallel SID_j \parallel DID_i \parallel DID_i^{new} \parallel e_i \parallel C_9)$. U_i then computes the session key $sk_u = \alpha C_6$, and checks $C_7 \stackrel{?}{=} h_1(C_2 \parallel C_6 \parallel sk_u \parallel DID_i \parallel SID_j)$. U_i terminates the session if U_i fails the verification phase. Otherwise, U_i computes $HPW_i^{new} = h(PW_i \parallel r_i^{new} \parallel R_i)$, $B_1^{new} = C_9 \oplus h(DID_i \parallel e_i) \oplus HPW_i^{new}$ 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 to verify 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 $C_{11} \stackrel{?}{=} h(ID_i \parallel e_i \parallel DID_i \parallel DID_i^{new})$. 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 $C_{12} \stackrel{?}{=} h(ID_i \parallel DID_i \parallel DID_i^{new} \parallel e_i \parallel C_9)$. 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 $Gen(B_i^{new}) = (R_i^{new}, P_{bi}^{new})$, $HPW_i^{new2} = h(PW_i^{new} \parallel r_i^{new} \parallel R_i^{new})$, $B1^{new2} = C_9 \oplus h(DID_i \parallel e_i) \oplus HPW_i^{new2}$ and $B_2^{new2} = h(ID_i \parallel R_i^{new} \parallel PW_i^{new}) \oplus r_i^{new}$. Finally, U_i substitutes $(B_1^{new2}, B_2^{new2}, P_{bi}^{new2})$ 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 $M_1 = \{C_1, C_2, C_3, C_4, Z_i, DID_i, SID_j\}$ 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 $r_{\mathcal{A}}^{new}$, $e_{\mathcal{A}}$, and $\alpha_{\mathcal{A}} \in [1, n-1]$, and selects a special sensor SID_j. $C_{1\mathcal{A}} = B_{1\mathcal{A}} \oplus HPW_{\mathcal{A}} \oplus e_{\mathcal{A}}$, $C_{2\mathcal{A}} = \alpha_{\mathcal{A}}P$, $C_{3\mathcal{A}} = h\,(e_{\mathcal{A}}) \oplus DID_i$, $Z_{\mathcal{A}} = ID_{\mathcal{A}} \oplus h\,(e_{\mathcal{A}} \parallel DID_{\mathcal{A}})$ and $C_{4\mathcal{A}} = h\,(ID_{\mathcal{A}} \parallel e_{\mathcal{A}} \parallel DID_{\mathcal{A}} \parallel DID_i \parallel C_{2\mathcal{A}} \parallel SID_j)$.
2. A forwards the login request message $M_{1\mathcal{A}} = \{C_{1\mathcal{A}}, C_{2\mathcal{A}}, C_{3\mathcal{A}}, C_{4\mathcal{A}}, Z_{\mathcal{A}}, DID_{\mathcal{A}}, SID_j\}$ to the gateway node GWN.
3. After receiving the login request message from $\mathcal{A}$, GWN computes $e_{\mathcal{A}} = C_{1\mathcal{A}} \oplus h\,(DID_{\mathcal{A}} \parallel x)$, $DID_i = C_{3\mathcal{A}} \oplus h\,(e_{\mathcal{A}})$and $ID_{\mathcal{A}} = Z_{\mathcal{A}} \oplus h\,(e_{\mathcal{A}} \parallel DID_{\mathcal{A}})$, and checks the validity of $ID_{\mathcal{A}}$ and $C_{4\mathcal{A}} \stackrel{?}{=} h\,(ID_{\mathcal{A}} \parallel e_{\mathcal{A}} \parallel DID_{\mathcal{A}} \parallel DID_i \parallel C_{2\mathcal{A}} \parallel SID_j)$. GWN then computes $c_j = h\,(SID_j \parallel x)$ and $C_{5\mathcal{A}} = h\,(c_j \parallel DID_j \parallel SID_j \parallel C_{2\mathcal{A}})$ and sends $M_{2\mathcal{A}} = \{C_{2\mathcal{A}}, C_{5\mathcal{A}}, DID_{\mathcal{A}}\}$ to S_j.
4. S_j checks $C_{5\mathcal{A}} \stackrel{?}{=} h\,(c_j \parallel DID_{\mathcal{A}} \parallel SID_j \parallel C_{2\mathcal{A}})$ with its identity SID_j. If this does not hold, S_j terminates the session. S_j then selects $\beta_{\mathcal{A}} \in [1, n-1]$ and computes $C_{6\mathcal{A}} = \beta_{\mathcal{A}}P$, $sk_s = \beta_{\mathcal{A}}C_{2\mathcal{A}}$, $C_{7\mathcal{A}} = h_1\,(C_{2\mathcal{A}} \parallel C_{6\mathcal{A}} \parallel sk_s \parallel DID_{\mathcal{A}} \parallel SID_j)$ and $C_{8\mathcal{A}} = h\,(DID_{\mathcal{A}} \parallel SID_j \parallel c_j)$. S_j sends $M_{3\mathcal{A}} = \{C_{6\mathcal{A}}, C_{7\mathcal{A}}, C_{8\mathcal{A}}\}$to GWN.
5. GWN tests $C_{8\mathcal{A}} \stackrel{?}{=} h\,(DID_{\mathcal{A}} \parallel SID_j \parallel c_j)$. If this does not hold, GWN terminates the session; otherwise, GWN calculates $C_{9\mathcal{A}} = h\,(DID_i \parallel x) \oplus h\,(DID_{\mathcal{A}} \parallel e_{\mathcal{A}})$ and $C_{10\mathcal{A}} = h\,(ID_{\mathcal{A}} \parallel SID_j \parallel DID_{\mathcal{A}} \parallel DID_i \parallel e_{\mathcal{A}} \parallel C_{9\mathcal{A}})$. Finally, GWN sends the message $M_{4\mathcal{A}} = \{C_{6\mathcal{A}}, C_{7\mathcal{A}}, C_{9\mathcal{A}}, C_{10\mathcal{A}}\}$ to attacker $\mathcal{A}$.
6. $\mathcal{A}$ calculates $h\,(DID_i \parallel x) = h\,(DID_{\mathcal{A}} \parallel e_{\mathcal{A}}) \oplus C_{9\mathcal{A}}$. 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 $h\,(DID_{\mathcal{A}} \parallel x)$ 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 $r_{\mathcal{A}}^{new}$, $e_{\mathcal{A}}$, and $\alpha_{\mathcal{A}} \in [1, n-1]$ and selects a particular sensor SID_j. Then, $\mathcal{A}$ calculates $DID_{\mathcal{A}}^{new} = h\,(ID_{\mathcal{A}} \parallel r_{\mathcal{A}}^{new})$, $C_{1\mathcal{A}} = B_{1\mathcal{A}} \oplus HPW_{\mathcal{A}} \oplus e_{\mathcal{A}}$, $C_{2\mathcal{A}} = \alpha_{\mathcal{A}}P$, $C_{3\mathcal{A}} = h\,(e_{\mathcal{A}}) \oplus DID_{\mathcal{A}}^{new}$, $Z_{\mathcal{A}} = ID_i \oplus h\,(e_{\mathcal{A}} \parallel DID_{\mathcal{A}})$ and $C_{4\mathcal{A}} = h\,(ID_i \parallel e_{\mathcal{A}} \parallel DID_{\mathcal{A}} \parallel DID_{\mathcal{A}}^{new} \parallel C_{2\mathcal{A}} \parallel SID_j)$. $C_{4\mathcal{A}}$ 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 $M_{1\mathcal{A}}$.
3. A forwards the login request message $M_{1\mathcal{A}} = \{C_{1\mathcal{A}}, C_{2\mathcal{A}}, C_{3\mathcal{A}}, C_{4\mathcal{A}}, Z_{\mathcal{A}}, DID_{\mathcal{A}}, SID_j\}$ to GWN.
4. After obtaining the message from the $\mathcal{A}$, GWN calculates $e_{\mathcal{A}} = C_{1\mathcal{A}} \oplus h\,(DID_{\mathcal{A}} \parallel x)$, $DID_{\mathcal{A}}^{new} = C_{3\mathcal{A}} \oplus h\,(e_{\mathcal{A}})$ and $ID_i = Z_{\mathcal{A}} \oplus h\,(e_{\mathcal{A}} \parallel DID_{\mathcal{A}})$, and checks the availability of ID_i and checks $C_{4\mathcal{A}} \stackrel{?}{=} h\,(ID_i \parallel e_{\mathcal{A}} \parallel DID_{\mathcal{A}} \parallel DID_{\mathcal{A}}^{new} \parallel C_{2\mathcal{A}} \parallel SID_j)$. 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 (.), h_1 (.), 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 $G_1 = G_1' \oplus HPW_i$, $G_2 = h\ (ID_i \parallel R_i \parallel PW_i) \oplus GID_i$ and $G_3 = h\ (ID_i \parallel GID_i)$. $\{G_1, G_2, G_3, h\ (.), h_1\ (.), P\}$ 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.

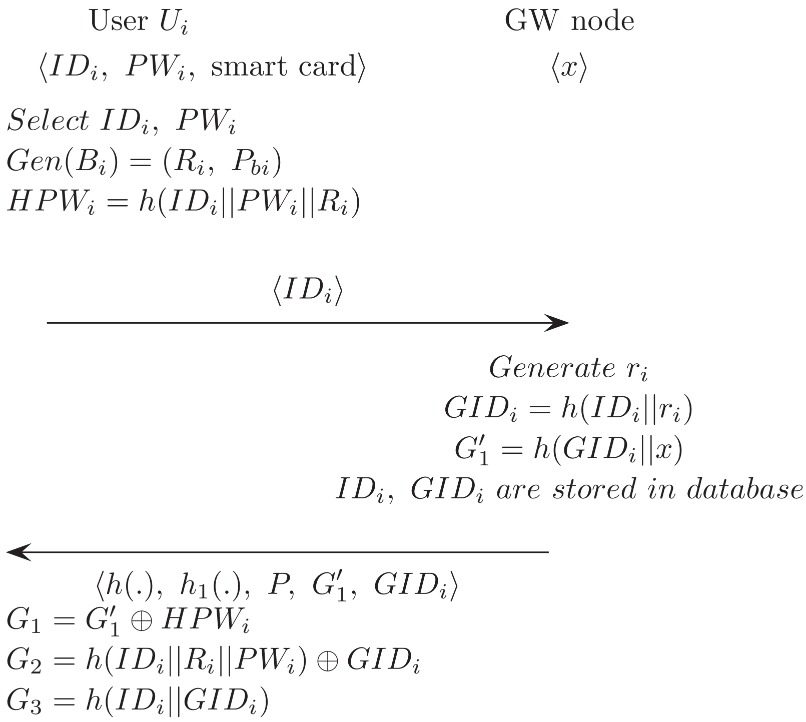

Figure 1. Registration phase of the proposed scheme.

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 $GID_i = G_2 \oplus h\ (ID_i \parallel R_i \parallel PW_i)$. U_i checks $h\ (ID_i \parallel GID_i) \stackrel{?}{=} G_3$. This allows U_i to verify whether it has come in correctly.
2. U_i generates e_i and α. U_i computes $HPW_i = h\ (ID_i \parallel PW_i \parallel R_i)$, $MU_1 = G_1 \oplus HPW_i \oplus e_i$, $MU_2 = \alpha P$ and $MU_3 = h\ (ID_i \parallel e_i \parallel GID_i \parallel MU_2 \parallel SID_j)$.
3. 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 $e_i = MU_1 \oplus h\ (GID_i \parallel x)$. GWN checks the validity of $MU_3 \stackrel{?}{=} h\ (ID_i \parallel e_i \parallel GID_i \parallel MU_2 \parallel SID_j)$. If it fails, the session will be terminated. Otherwise, GWN computes $c_j = h\ (SID_j \parallel x)$ and $MG_1 = h\ (c_j \parallel GID_i \parallel SID_j \parallel MU_2)$. When the operation has finished, GWN sends the message $M_2 = \{MU_2, MG_1, GID_i\}$ to S_j.
2. S_j checks $MG_1 \stackrel{?}{=} h\ (c_j \parallel GID_i \parallel SID_j \parallel MU_2)$ 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 $sk_s = \beta MU_2$, $MS_2 = h_1\ (MU_2 \parallel MS_1 \parallel sk_s \parallel GID_i \parallel SID_j)$ and $MS_3 = h\ (GID_i \parallel SID_j \parallel c_j)$. It sends message $M_3 = \{MS_1, MS_2, MS_3\}$ when all operations have finished.
3. GWN checks $MS_3 \stackrel{?}{=} h\ (GID_i \parallel SID_j \parallel c_j)$. If it is wrong, the session will be stopped. Otherwise, GWN generates r_i^{new} and calculates $GID_i^{new} = h\ (ID_i \parallel r_i^{new})$, $MG_2 = h\ (GID_i^{new} \parallel x) \oplus h\ (GID_i \parallel e_i)$, $MG_3 = h\ (ID_i \parallel SID_j \parallel GID_i \parallel GID_i^{new} \parallel e_i \parallel MG_2)$ and $MG_4 = h\ (e_i) \oplus GID_i^{new}$. Finally, GWN sends the message $M_4 = \{MS_1, MS_2, MG_2, MG_3, MG_4\}$ to U_i.
4. U_i computes $GID_i^{new} = MG_4 \oplus h\ (e_i)$ and checks $MG_3 \stackrel{?}{=} h\ (ID_i \parallel SID_j \parallel GID_i \parallel GID_i^{new} \parallel e_i \parallel MG_2)$. If not, the session will be stopped. U_i computes $sk_u = \alpha MS_1 = \alpha\beta P$ and checks $MS_2 \stackrel{?}{=} h_1\ (MU_2 \parallel MS_1 \parallel sk_u \parallel GID_i \parallel SID_j)$. If it is wrong, U_i will stop the session.

5. U_i computes $G_1^{new} = MG_2 \oplus h\ (GID_i \parallel e_i) \oplus HPW_i$, $G_2^{new} = G_2 \oplus GID_i \oplus GID_i^{new}$ and $G_3^{new} = h\ (ID_i \parallel GID_i^{new})$. 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.

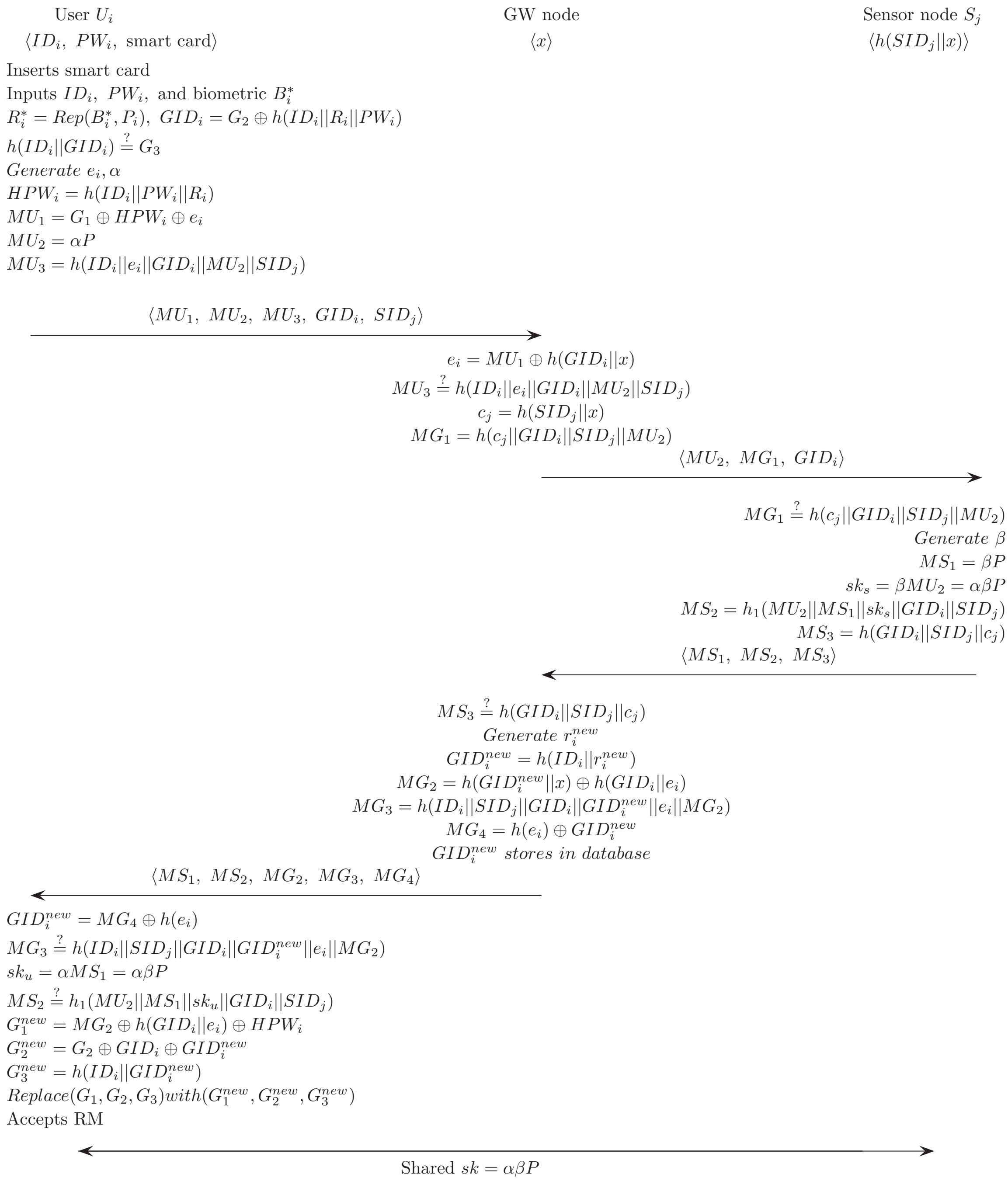

Figure 2. Login and authentication phase of the proposed scheme.

5.4. Password and Biometrics Change Phase

1. U_i inputs ID_i, PW_i and B_i'. The smart card executes $Rep\ (B_i', P_{bi}) = R_i$ and $GID_i = G_2 \oplus h\ (ID_i \parallel R_i \parallel PW_i)$. U_i checks $h\ (ID_i \parallel GID_i) \stackrel{?}{=} G_3$. 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: $Gen\ (B_i^{new}) = (R_i^{new}, P_{bi}^{new})$, $HPW_i^{new2} = h\ (ID_i \parallel PW_i^{new} \parallel R_i^{new})$, $G_1^{new2} =$

$G_1 \oplus HPW_i \oplus HPW_i^{new2}$, $G_2^{new2} = G_2 \oplus h\,(ID_i \parallel R_i \parallel PW_i) \oplus h\,(ID_i \parallel R_i \parallel PW_i^{new2})$. Finally, U_i substitutes (G_1^{new2}, G_2^{new2}, P_{bi}^{new}) 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.

```
(*—-Ui process—-*)
let Ui =
let HPWi = h(concat(concat(IDi, PWi), Ri)) in
out(cha,(IDi));
in(cha,(XGIDi:bitstring));
let G1' = h(concat(XGIDi, x)) in
let G1 = xor(G1', HPWi) in
let G2 = xor(h(concat(concat(IDi, Ri), PWi)), XGIDi) in
let G3 = h(concat(IDi, XGIDi)) in
event beginUi(IDi);
new ei:bitstring;
new alpha:bitstring;
let GIDi = xor(G2, h(concat(concat(IDi, Ri), PWi))) in
if h(concat(IDi, XGIDi)) = G3 then
let HPWi = h(concat(concat(IDi, PWi), Ri)) in
let MU1 = xor(xor(G1, HPWi), ei) in
let MU2 = mult(alpha, P) in
let MU3 = h(concat(concat(IDi, ei), concat(concat(XGIDi, MU2), SIDj))) in
out(chc,(MU1, MU2, MU3, GIDi, SIDj));
in(chc,(XXMS1:bitstring, XXMS2:bitstring,
XMG2:bitstring, XMG3:bitstring, XMG4:bitstring));
let GIDinew = xor(XMG4, h(ei)) in
if XMG3 = h(concat(concat(IDi, SIDj),
concat(concat(GIDi, GIDinew), concat(ei, XMG2)))) then
let sku = mult(alpha, XXMS1) in
if XXMS2 = h1(concat(concat(MU2, XXMS1),
concat(concat(sku, GIDi), SIDj))) then
let G1new = xor(XMG2, xor(h(concat(GIDi, ei)), HPWi)) in
let G2new = xor(G2, xor(GIDi, GIDinew)) in
let G1 = G1new in
let G2 = G2new in
event endUi(IDi).
```

Table 4. GWN protocol.

```
(*—-GWN process—-*)
let GWN =
in(cha, (XIDi:bitstring));
new ri:bitstring;
let GIDi = h(concat(XIDi, ri)) in
let G1' = h(concat(GIDi, x)) in
out(cha, (GIDi));
in(chc, (XMU1:bitstring, XMU2:bitstring, XMU3:bitstring, XGIDi:bitstring, XSIDj:bitstring));
event beginGWN(IDg);
let ei = xor(XMU1,h(concat(XGIDi, x))) in
if XMU3 = h(concat(concat(XIDi, ei),
concat(concat(XGIDi, XMU2), XSIDj))) then
let cj = h(concat(XSIDj, x)) in
let MG1 = h(concat(concat(cj, XGIDi), concat(XSIDj, XMU2))) in
out(chb, (XMU2, MG1, XGIDi));
in(chb, (XMS1:bitstring, XMS2:bitstring,
XMS3:bitstring));
if XMS3 = h(concat(concat(XGIDi, XSIDj), cj)) then
new rinew:bitstring;
let GIDinew = h(concat(XIDi, rinew)) in
let MG2 = xor(h(concat(GIDinew, x)), h(concat(XGIDi, ei))) in
let MG3 = h(concat(concat(XIDi, XSIDj), concat(concat(XGIDi, GIDinew), concat(ei, MG2)))) in
let MG4 = xor(h(ei), GIDinew) in
out(chc, (XMS1, XMS2, MG2, MG3, MG4));
event endGWN(IDg).
```

Table 5. S_j protocol.

```
(*—-Sj process—-*)
let Sj =
in(chb, (XXMU2:bitstring, XMG1:bitstring, XXGIDi:bitstring));
event beginSj(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 endSj(SIDj).
```

Table 6. Queries.

```
(*—-queries—-*)

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(endSj(id)) ==> inj-event(beginSj(id)).

process
((!Ui) | (!GWN) | (!Sj))
```

Table 7. Output of queries.

```
RESULT inj-event(endSj(id)) ==> inj-event(beginSj(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_i^{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 (.), h_1 (.), 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_is 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 β or α when knowing αP or βP with $\alpha\beta P$. Neither β nor α 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, $e_i = MU_1 + h\ (GID_i||x)$ is calculated in order to compare the MU_3 and $h\ (\ ID_i||e_i||GID_i||MU_2||SIDj_)$. 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 αP and βP through $\alpha\beta P$, we can not find $\alpha\beta P$ through αP and βP, and we can not find α through P and αP.

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.

Table 9. Notations of time symbol.

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	$6T_h + 2T_m + 1T_e$ = 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 α and a string β
Rep	A function that restore β to α and any vector B^* close to B
B	A vector with biometric information
B^*	Any vector B^* close to B
GWN	Gateway node
ProVerif	An analysis tool for protocol verification

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. Moon, J.; Choi, Y.; Kim, J.; Won, D. An improvement of robust and efficient biometrics based password authentication scheme for telecare medicine information systems using extended chaotic maps. *J. Med. Syst.* **2016**, *40*, 70. [CrossRef] [PubMed]
11. He, D.; Gao, Y.; Chan, S.; Chen, C.; Bu, J. An enhanced two-factor user authentication scheme in wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* **2010**, *10*, 361–371.
12. 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.
13. 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]
14. Xue, K.; Ma, C.; Hong, P.; Ding, R. A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. *J. Netw. Comput. Appl.* **2013**, *36*, 316–323. [CrossRef]
15. 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]
16. 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]
17. Das, A.K. A secure and effective biometricbased user authentication scheme for wireless sensor networks using smart card and fuzzy extractor. *Int. J. Commun. Syst.* **2017**, *30*, e2933. [CrossRef]
18. Das, A.K. A secure and efficient user anonymity-preserving three-factor authentication protocol for large-scale distributed wireless sensor networks. *Wirel. Pers. Commun.* **2015**, *82*, 1377–1404. [CrossRef]
19. Miller, V. Uses of Elliptic Curves in Cryptography. In *Advances in Cryptology Crypto*; Springer: Berlin, Germany, 1986; Volume 218, pp. 417–426.
20. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209. [CrossRef]
21. Dodis, Y.; Kanukurthi, B.; Katz, J.; Smith, A. Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Trans. Inf. Theory* **2013**, *58*, 6207–6222. [CrossRef]
22. Dodis, Y.; Reyzin, L.; Smith, A. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004; pp. 523–540.
23. 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]
24. 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]
25. Park, Y.; Park, Y. Three-factor user authentication and key agreement using elliptic curve cryptosystem in wireless sensor networks. *Sensors* **2016**, *16*, 2123. [CrossRef] [PubMed]
26. 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]