sensors

Article

A Lightweight Anonymous Authentication Protocol with Perfect Forward Secrecy for Wireless Sensor Networks

Ling Xiong [1], Daiyuan Peng [1], Tu Peng [2], Hongbin Liang [3,*] and Zhicai Liu [4]

1 School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China; lingdonghua99@163.com (L.X.); dypeng@swjtu.edu.cn (D.P.)
2 School of Software, Beijing Institute of Technology, Beijing 100081, China; pengtu@bit.edu.cn
3 School of Transportation and Logistics, Southwest Jiaotong University, Chengdu 611756, China
4 School of Computer and Software Engineering, Xihua University, Chengdu 610039, China; idle@gmail.com
* Correspondence: hbliang@swjtu.edu.cn; Tel.: +86-28-8760-0740

Received: 27 September 2017; Accepted: 16 November 2017; Published: 21 November 2017

Abstract: Due to their frequent use in unattended and hostile deployment environments, the security in wireless sensor networks (WSNs) has attracted much interest in the past two decades. However, it remains a challenge to design a lightweight authentication protocol for WSNs because the designers are confronted with a series of desirable security requirements, e.g., user anonymity, perfect forward secrecy, resistance to de-synchronization attack. Recently, the authors presented two authentication schemes that attempt to provide user anonymity and to resist various known attacks. Unfortunately, in this work we shall show that user anonymity of the two schemes is achieved at the price of an impractical search operation—the gateway node may search for every possible value. Besides this defect, they are also prone to smart card loss attacks and have no provision for perfect forward secrecy. As our main contribution, a lightweight anonymous authentication scheme with perfect forward secrecy is designed, and what we believe the most interesting feature is that user anonymity, perfect forward secrecy, and resistance to de-synchronization attack can be achieved at the same time. As far as we know, it is extremely difficult to meet these security features simultaneously only using the lightweight operations, such as symmetric encryption/decryption and hash functions.

Keywords: mutual authentication; user anonymity; wireless sensor networks; strand space model

1. Introduction

Wireless sensor networks (WSNs) have gained a great deal of attention from researchers in the academic and industrial field mainly because of two reasons: first, they consist of a large number of resource-constrained sensor nodes, which are deployed randomly in a target region [1], and second, they can be widely used in various kinds of applications, such as healthcare monitoring [2], environment sensing [3], industrial monitoring [4], etc. Generally, WSNs are developed to monitor physical or environmental conditions, such as temperature, humidity, sound, etc. and collect real-time information about these conditions. In many applications [5–7], external users need to access to this real-time information from the sensor nodes. Figure 1 describes a way for real-time information access in WSNs. For example, using a WSN in the healthcare environment, the patient's real-time information such as temperature, blood pressure, and pulse rate, will be collected by sensor nodes. Then, legitimate medical workers are able to access these data directly from the sensor nodes.

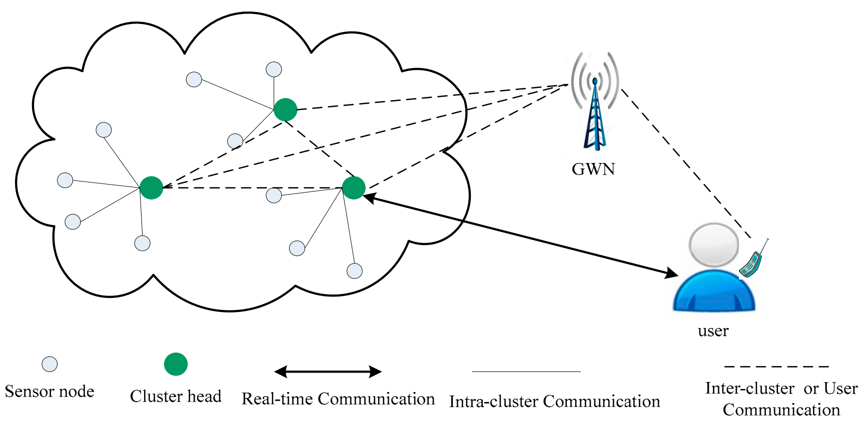

GWN
user
Sensor node
Cluster head
Real-time Communication
Intra-cluster Communication
Inter-cluster or User Communication

Figure 1. Real-time data access in WSNs.

Although it seems appealing for users to access the real-time data from the sensor nodes, user authentication has been a critical issue in WSNs due to their frequent use in unattended and hostile environments [8]. Because many applications for WSNs operate in such environments, such as battlefields, a malicious adversary could easily control the communication channel, i.e., he/she would be able to eavesdrop, insert, block, and alter the transmitted data. Thus, WSNs are subject to various types of attacks. To ensure that only authorized users can access the reliable sensor nodes and to protect the real-time information, it is indispensable to achieve mutual authentication and establish a session key between the user and the sensor node. Nowadays, there are mainly three ways to accomplish authenticated key establishment scheme in WSNs [9].

- The first and the simplest solution for the authenticated key establishment is a shared symmetric key between the user and the sensor node. In this case, if a WSN has n sensor nodes and m users, each sensor node needs to store m symmetric keys, each user needs to store n symmetric keys, and the WSN needs to establish nm symmetric keys.
- Secondly, using public key cryptography, like ECC [10], RSA [11] or ElGamal [12], is another approach to complete authenticated key establishment.
- Third, the user and the sensor node can achieve mutual authentication and establish a session key through a trust gateway node (GWN) [13–30]. In this case, both the user and the sensor node need to share only a single key with the GWN. The GWN can help the user and the sensor authenticate each other and distribute a shared secret session key at each session. After this phase, the user can use this session key to access the real-time data from the desired sensor node without involving the GWN.

Obviously, the first method does not scale well, and the second way using public key cryptography primitives may tend to be resource intensive because most of them are based on the large integer. Hence, the authenticated key establishment scheme with the help of the GWN is even more admired owing to limited computation and communication resources, capability, bandwidth of sensor nodes. Additionally, identity masquerade and identity tracing have become common attacks in WSNs, which will cause the problem of identity privacy. Hence, there is a growing demand to achieve anonymous authentication in WSNs. Besides, since the sensor node is unattended, the long-term key of the sensor node may be compromised by an adversary. In this case, the previous session keys will be in danger. To address it, perfect forward secrecy should be considered. Therefore, anonymous authentication schemes with perfect forward secrecy for WSNs should be designed by using only the lightweight cryptographic primitives, such as symmetric key encryption/decryption and hash functions.

Many anonymous authentication schemes using lightweight cryptographic primitive have been proposed for WSNs in the past several years. However, as far as we know, most of them cannot consider perfect forward secrecy or suffer from de-synchronization attack. In this work, we design a lightweight authentication scheme for WSNs, which can achieve user anonymity, perfect forward secrecy, and resistance to de-synchronization attack at the same time.

1.1. Related Works

In some applications for WSNs, such as real-time healthcare monitoring, traffic control monitoring, and military surveillance, external users are interested in accessing real-time data directly from desired sensor nodes without involving the GWN. User authentication is an essential security measure for the user to be first authorized to the GWN as well as the sensor nodes before granting access to the real-time data. To achieve user authentication in WSNs, hundreds of schemes have been proposed in the last decade, such as remarkable schemes [13–30]. In 2006, Wong et al. [13] designed a dynamic strong-password authentication scheme for WSNs using lightweight operations, such as one-way hash function and XOR operations. But later, Das [14] pointed out that Wong et al.'s scheme is vulnerable to replay attack and stolen-verifier attack. In order to address these issues, Das [14] presented a two-factor authenticated key establishment scheme for WSNs, which claimed to provide strong authentication and resist various kinds of attacks. Unfortunately, a series of articles [15–19] have indicated that Das's scheme [14] has still some drawbacks and flaws, such as susceptibility to privileged-insider attacks, smart card loss attacks, and parallel session attacks.

Although the abovementioned schemes [15–19] have much better performance than Das' scheme [14], they are still prone to several security flaws, such as smart card loss attacks and forgery attacks. In 2012, Das et al. [20] developed a better scheme to solve these weaknesses. However, the security of Das et al.'s new scheme was not satisfactory, because of its vulnerability to some attacks [21–23]. After that, Xue et al. [24] designed a temporal-credential based authenticated key agreement scheme for WSNs using the hash function and XOR operations, which claimed to provide identity and password protection, and resiliency of smart card loss attacks. Unfortunately, Jiang et al. [25] described how the Xue et al.'s scheme [24] was insecure against identity guessing attack, privileged insider attack, tracking attack and smart card loss attack. They proposed an efficient two-factor user authentication scheme with unlinkability property in WSNs. The unlinkability pseudonym identity can help the GWN to quickly search the exact communicating user. Besides, it is able to provide user anonymity and resistance to smart card loss attack. In 2016, Das [26] proposed an enhanced authentication scheme based on Jiang et al.'s scheme [25]. He insisted that their scheme can provide higher security level than other schemes.

To the designers' disappointment, both Jiang et al.'s scheme and Das' scheme have been found vulnerable to the desynchronization attack [31]. Most recently, Gope and Hwang [27] proposed a realistic authentication scheme for WSNs, which can ensure various kinds of imperative security properties like mutual authentication, user anonymity, perfect forward secrecy, etc. For the communication between the user and the GWN, Gope and Hwang's scheme employed a set of unlinkable shadow-IDs and emergency keys to prevent de-synchronization attack [31]. This is the preferred way to solve the de-synchronization attack. However, for the communication between the GWN and the sensor node, if the adversary blocks the response message flow from the sensor node, the communication will be lost in synchronization. Thus the sensor node needs to ask the GWN for the new secret shared key. Besides, we observe that their scheme also cannot resist against known session-specific temporary information attack [28]. When the session-specific temporary information N_u is disclosed to the adversary, it is obvious that K_{ug} can be resumed from transmitted message $N_x = N_u \oplus K_{ug}$. Then, the adversary generates his own Ts^*_{ugnew} and the session key SK^*, and computes $Ts^* = h(K_{ug}||ID_U||N_u) \oplus Ts^*_{ugnew}$, $SK^{*\prime\prime} = h(K_{ug}||ID_U||N_u) \oplus SK^*$, $V_4 = h(SK^{*\prime\prime}||N_u||Ts^*||K_{ug})$, where ID_U is the identity of the user, which can be off-line guessed by transmitted messages Ks_{ug} and $AID_U = h(ID_U||K_{ug}||N_u||Ks_{ug})$. Thus, the adversary can successfully forge a legal gateway node authentication message and get the session key.

In the same year, the authors of [28,29] proposed a lightweight authentication scheme, which uses a 'dynamic ID technique' to achieve user anonymity and is secure in resisting known session-specific temporary information attack. Unfortunately, we find that the two schemes are insecure against smart card loss attacks. Besides, both schemes have two design flaws, including impractical GWN search operation and no provision for perfect forward secrecy.

On the other hand, perfect forward secrecy is an important security property for authenticated schemes. Unfortunately, to the best of our knowledge, most of the authentication schemes only using lightweight cryptographic primitives cannot provide perfect forward secrecy (e.g., the recent pertinent authentication schemes [24–26,28–30]). Although Mir et al. [30] claimed that their scheme is secure in perfect forward secrecy, we find out it is still prone to forward secrecy attack. In this scheme, the authors proved that the session key $SK = h(K_i || K_j || ID_i || SID_i || T_1)$ is secure under the assumption that the adversary A does not obtain the identity ID_i of the user. However, if GWN's secret key d is compromised, A can offline guess ID_i through the transmitted message M as below. A guesses a candidate ID'_i and computes $X'_i = h(ID'_i || d)$, $ID_i || K_i || T_1 || H_i = D_{X'i}(M)$. A checks whether ID'_i and ID_i are equivalent. If they are equal, A can obtain the correct ID_i. Otherwise, A repeats this operation until the correct ID_i is obtained. Hence, Mir et al.'s scheme is unable to achieve perfect forward secrecy. Several articles [32–34] pointed out that it is intrinsically unable to provide perfect forward secrecy in the scheme that does not employ public-key primitives. To the best of our knowledge, some schemes [27] tried to address this issue using the one-time hash chain technique. However, it may cause de-synchronization attack because the hash chain value will be updated after each successful session.

1.2. Motivation and Contributions

Two previously-thought sound schemes [28,29] use 'dynamic ID technique' to achieve user anonymity at the price of the impractical exhaustive search operations. The reason is that users' real identities are encoded into dynamic identities, no one is able to get the identity information of the user without the secret key. When the user wants to access the WSNs systems, it is difficult for the GWN to tell apart the real identity of the user. As a result, the GWN needs to search for every possible parameter to figure out the exact user. Generally, to address this, a pseudonym identity method [25,26] is used to help the GWN to read the correct information from the user information table. In this way, both of the user and the GWN store a randomly generated pseudonym identity, which is updated after each successful session. Since the pseudonym identity is different at each session, the adversary cannot track a specific user. However, Wang et al. [31] pointed out the scheme using pseudonym identity may easily suffer from the de-synchronization attacks, which may render the scheme completely unusable unless the user or the sensor node re-registers.

To the best of our knowledge, the hash chain technique can be employed to ensure perfect forward secrecy for lightweight cryptographic protocols [27]. However, like the pseudonym identity method, both communicating parties need to update their shared one-time hash chain value after completion of each session. Thus, the technique may also cause the de-synchronization attack.

Motivated by the above facts, we construct a new efficient authentication scheme for WSNs using the pseudonym identity method and one-time hash chain technique to achieve user anonymity and perfect forward secrecy. For the communication between the user and the GWN, the back-end of GWN stores two pseudonym identities PID_{i0} and PID_{i1} to resist against de-synchronization attack. PID_{i0} stores the value of the new pseudonym identity. PID_{i1} has two functions: the one is storing the value of the old pseudonym identity, the other is a tag for updating hash chain. If $PID_{i1} = \perp$, it means that the value of hash chain has updated in the previous session. Otherwise, the value of hash chain does not change, where $\perp$ denotes null. For the communication between the GWN and the sensor node, serial number technique is used to resist against de-synchronization attack.

Altogether, in this paper, we analyze the security of two representative schemes [28,29] for WSNs and show their vulnerability to smart card loss attack, impractical GWN search operation and no provision for perfect forward secrecy. To overcome these weaknesses, we design a lightweight anonymous authentication protocol for WSNs based on the one-time hash chain and pseudonym identity. The main contributions of our scheme are summarized as follows:

(1) The proposed scheme is resilient to various kinds of known attacks, such as de-synchronization attack, known session-specific temporary information attack;
(2) The proposed scheme can provide mutual authentication, user anonymity, and perfect forward secrecy, etc.
(3) The proposed scheme uses lightweight cryptographic primitives, such as symmetric encryption/decryption and hash functions. It is very suitable for the resource constrained sensor nodes.

1.3. Adversary Model

An adversary A has five goals. The first is that A can successfully impersonate the user U_i authenticating to GWN. The second is that A can successfully impersonate GWN authenticating to U_i. The third is that A can successfully impersonate the sensor node S_{nj} authenticating to GWN. The fourth is that A can successfully impersonate GWN authenticating to S_{nj}. And the last is that A can obtain the session key among U_i, GWN and S_{nj}. We assume that A is a probabilistic polynomial time attacker, and the feasible attacks are summarized as follows:

- ➢ A can control the channel among U_i, GWN and S_{nj}. It means that A can eavesdrop, insert, block, and alter the transmitted messages through the public communication channel.
- ➢ A can obtain one of the two authentication factors, smart card or password. If A has obtained the smart card, he can extract the secret value in the smart card and has the capability of enumerating identity and password space $|D_{ID}{*}D_{PW}|$.
- ➢ A may be another legitimate but malicious user in the system.
- ➢ A may be a legitimate but malicious sensor node.

1.4. Notations

All the notations mentioned in two related schemes and our proposed scheme are defined in Table 1.

Table 1. Notations.

Notation	Descriptions
U_i	The user
S_{nj}	The sensor node
GWN	The gateway node
SC	The smart card
ID_i,PW_i	Unique identity and password of U_i
SID_j	Unique identity of S_{nj}
ID_{GWN}	Unique identity of GWN
PID	Pseudonym identity
PID_i	Pseudonym identity of U_i in the user side
TID_i	A random number of U_i generated in the GWN
x	The secret key of GWN
x_i	The shared secret key between GWN and U_i
K_{GWN_S}	The shared secret key between GWN and Sn_j
b_i	A random number generate by U_i
SK	The session key
E_K,D_K	Encryption/Decryption using the symmetric key K
h,h_0,h_1,h_3	One-way hash function
h_2	One-way hash function, h_2:{0,1}*→{0,1,...,1023}
T,T_1,T_2,T_3,T_4	Current timestamp
||	String concatenation operation
$\oplus$	XOR operation

1.5. Organization of the Paper

This paper takes two relate schemes [28,29] as case studies, we present a concrete attack to show that the two schemes are insecure against the smart card loss attack. Besides, we also show both schemes have two design flaws, including impractical GWN search operation and no provision for perfect forward secrecy. Then, we put forward a new way to deal with de-synchronization attack and design an efficient anonymous authentication scheme with perfect forward secrecy for WSNs.

The rest of this paper is organized as follows: Section 2 reviews two related schemes for WSNs. Section 3 presents the detailed procedure of the proposed scheme. Section 4 gives security analysis of our scheme. The computation and communication costs analysis of the proposed scheme are discussed in Section 5. Finally, Section 6 concludes this paper.

2. Review of Two Related Schemes

This section will describe two related authenticated key establishment schemes for WSNs, which are Lu et al.'s scheme [28] and Jung et al.'s scheme [29]. The reason for choosing these two schemes is that they are the typical representations of recent schemes in WSNs which have the security flaws in smart card loss attack, impractical GWN search operation and no provision for perfect forward secrecy. First, we will give briefly review of two schemes. Later on, the detailed weaknesses of the two schemes will be described.

2.1. Review of Lu et al.'s Scheme

Lu et al.'s authentication scheme [28] is shown in Figure 2. This scheme consists of four phases: the user registration, the sensor node registration, login and authentication, password change.

2.1.1. User Registration

Step 1: A new user U_i selects the identity ID_i and the password PW_i, generates a random number b_i. Then U_i computes $C_i = h(PW_i || b_i)$, U_i transmits $\{ID_i, C_i\}$ to GWN through a secure channel.

Step 2: Upon receipt of the message, the GWN computes $A_i = h(h(ID_i) || C_i)$, $B_i = h(TID_i || x) \oplus C_i$, $M_i = h(ID_i || x) \oplus h(h(ID_i) \oplus C_i)$. After that, GWN stores $\{TID_i\}$ in its memory, and stores $\{A_i, B_i, M_i\}$ into smart card SC. Finally, GWN sends SC to U_i via a private channel.

Step 3: After receiving SC from GWN, U_i stores b_i into SC.

2.1.2. Sensor Node Registration

Step 1: A new sensor node S_{nj} selects identity SID_j and transmits $\{SID_j\}$ to GWN through a secure channel.

Step 2: The GWN computes $A_j = h(SID_j \oplus K_{GMN\_S})$, and returen it to S_{nj} after storing $\{SID_j, A_j\}$ into its memory.

Step 3: After receiving SID_j, A_j from GWN, S_{nj} stores them into its memory as the secret.

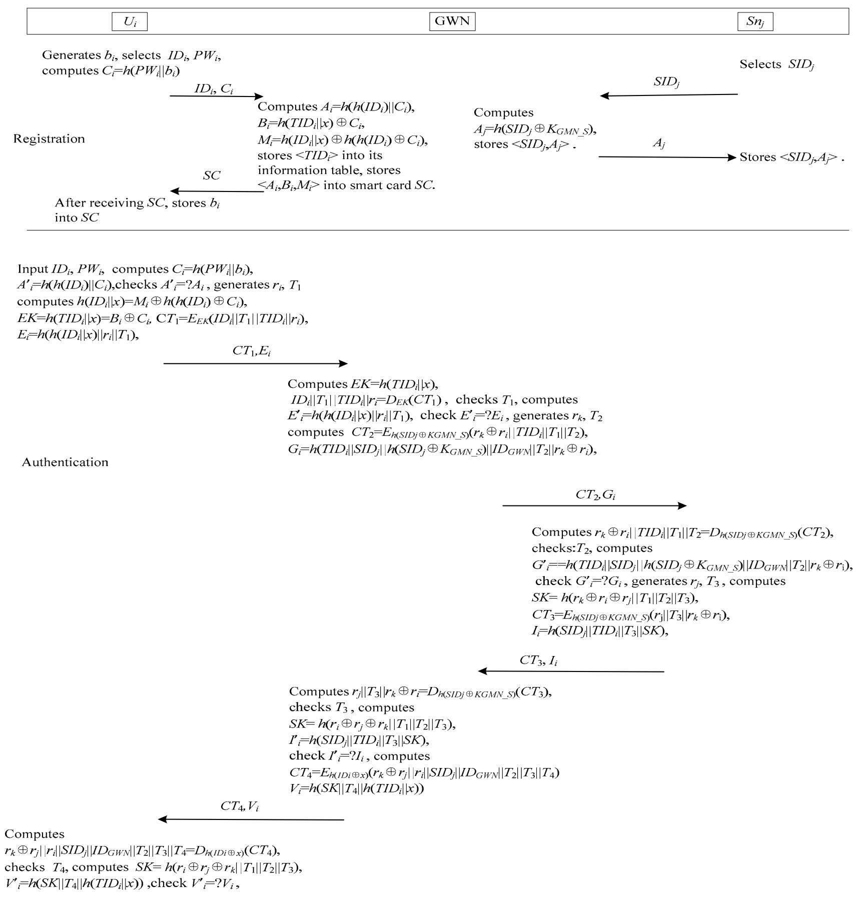

U_i
GWN
Sn_j
Generates b_i, selects ID_i, PW_i, computes C_i=$h(PW_i||b_i)$
ID_i, C_i
Computes A_i=$h(h(ID_i)||C_i)$, B_i=$h(TID_i||x) \oplus C_i$, M_i=$h(ID_i||x) \oplus h(h(ID_i) \oplus C_i)$, stores <$TID_i$> into its information table, stores <A_i,B_i,M_i> into smart card SC.
SC
After receiving SC, stores b_i into SC
Registration
Selects SID_j
SID_j
Computes A_j=$h(SID_j \oplus K_{GMN\_S})$, stores <$SID_j$,$A_j$> .
A_j
Stores <SID_j,A_j> .
Input ID_i, PW_i, computes C_i=$h(PW_i||b_i)$, A'_i=$h(h(ID_i)||C_i)$,checks A'_i=?A_i , generates r_i, T_1 computes $h(ID_i||x)$=$M_i \oplus h(h(ID_i) \oplus C_i)$, EK=$h(TID_i||x)$=$B_i \oplus C_i$, CT_1=$E_{EK}(ID_i||T_1||TID_i||r_i)$, E_i=$h(h(ID_i||x)||r_i||T_1)$,
CT_1,E_i
Computes EK=$h(TID_i||x)$, $ID_i||T_1||TID_i||r_i$=$D_{EK}(CT_1)$, checks T_1, computes E'_i=$h(h(ID_i||x)||r_i||T_1)$, check E'_i=?E_i , generates r_k, T_2 computes CT_2=$E_{h(SIDj \oplus KGMN\_S)}(r_k \oplus r_i||TID_i||T_1||T_2)$, G_i=$h(TID_i||SID_j||h(SID_j \oplus K_{GMN\_S})||ID_{GWN}||T_2||r_k \oplus r_i)$,
Authentication
CT_2,G_i
Computes $r_k \oplus r_i||TID_i||T_1||T_2$=$D_{h(SIDj \oplus KGMN\_S)}(CT_2)$, checks:$T_2$, computes G'_i==$h(TID_i||SID_j||h(SID_j \oplus K_{GMN\_S})||ID_{GWN}||T_2||r_k \oplus r_i)$, check G'_i=?G_i , generates r_j, T_3 , computes SK= $h(r_k \oplus r_i \oplus r_j||T_1||T_2||T_3)$, CT_3=$E_{h(SIDj \oplus KGMN\_S)}(r_j||T_3||r_k \oplus r_i)$, I_i=$h(SID_j||TID_i||T_3||SK)$,
CT_3, I_i
Computes $r_j||T_3||r_k \oplus r_i$=$D_{h(SIDj \oplus KGMN\_S)}(CT_3)$, checks T_3 , computes SK= $h(r_i \oplus r_j \oplus r_k||T_1||T_2||T_3)$, I'_i=$h(SID_j||TID_i||T_3||SK)$, check I'_i=?I_i , computes CT_4=$E_{h(IDi \oplus x)}(r_k \oplus r_j||r_i||SID_j||ID_{GWN}||T_2||T_3||T_4)$ V_i=$h(SK||T_4||h(TID_i||x))$
CT_4,V_i
Computes $r_k \oplus r_j||r_i||SID_j||ID_{GWN}||T_2||T_3||T_4$=$D_{h(IDi \oplus x)}(CT_4)$, checks T_4, computes SK= $h(r_i \oplus r_j \oplus r_k||T_1||T_2||T_3)$, V'_i=$h(SK||T_4||h(TID_i||x))$,check V'_i=?V_i ,

Figure 2. The authentication and key agreement phase of Lu et al.'s scheme.

2.1.3. Login

When a user U_i desires the WSNs services, he/she needs to achieve mutual authenticate with GWN and S_{nj}. As shown in Figure 2, the process of mutual authentication is described as follows.

Step 1: U_i inputs ID_i and PW_i into the smart card SC. SC computes $C_i = h(PW_i || b_i)$, $A'_i = h(h(ID_i) || C_i)$, and compares A'_i with the stored value A_i. If they are not equal, SC terminates the session. Otherwise, SC believes U_i as a legitimate user. Next, SC generates a random number r_i, and computes $h(ID_i || x) = M_i \oplus h(h(ID_i) \oplus C_i)$, $EK = h(TID_i || x) = B_i \oplus C_i$, $CT_1 = E_{EK}(ID_i || T_1 || TID_i || r_i)$, $E_i = h(h(ID_i || x) || r_i || T_1)$, where T_1 is the timestamp. Finally, SC sends the login request $\{CT_1, E_i\}$ to GWN through the public channel.

Step 2: After receiving the login messages, the GWN computes $EK = h_1(TID_i || x)$, $ID_i || T_1 || TID_i || r_i = D_{EK}(CT_1)$. Then, the GWN checks the timestamp T_1, computes $E'_i = h(h(ID_i || x) || r_i || T_1)$, and checks whether E'_i matches with the received E_i. If it does not hold, GWN terminates the session. Otherwise, the GWN generates a random numnber r_k, and computes $CT_2 = E_{h(SIDj \oplus KGMN\_S)}(r_k \oplus r_i || TID_i || T_1 || T_2)$, $G_i = h(TID_i || SID_j || h(SID_j \oplus K_{GMN\_S}) || ID_{GWN} || T_2 || r_k \oplus r_i)$, where T_2 is the timestamp. Finally, the GWN sends $\{CT_2, G_i\}$ to the sensor node S_{nj} that U_i wants to interact with via the public channel.

Step 3: Upon receiving the messages $\{CT_2, G_i\}$ from GWM, S_{nj} at first computes $r_k \oplus r_i || TID_i || T_1 || T_2 = D_{h(SIDj \oplus KGMN\_S)}(CT_2)$. Then, S_{nj} checks the timestamp T_2, computes $G'_i = h(TID_i || SID_j || h(SID_j \oplus K_{GMN\_S}) || ID_{GWN} || T_2 || r_k \oplus r_i)$, and checks whether G'_i matches with the received G_i. If it does not hold, S_{nj} terminates the session. Otherwise, the S_{nj} generates a random numnber r_j, and computes $SK = h(r_k \oplus r_i \oplus r_j || T_1 || T_2 || T_3)$,

$CT_3 = E_{h(SIDj \oplus KGMN\_S)}(r_j || T_3 || r_k \oplus r_i)$, $I_i = h(SID_j || TID_i || T_3 || SK)$, where T_3 is the timestamp. Finally, S_{nj} transmits $\{CT_3, I_i\}$ to GWN.

Step 4: GWN first computes $r_j || T_3 || r_k \oplus r_i = D_{h(SIDj \oplus KGMN\_S)}(CT_3)$. Then, the GWN checks the timestamp T_3, and computes $SK = h(r_k \oplus r_i \oplus r_j || T_1 || T_2 || T_3)$, $I'_i = h(SID_j || TID_i || T_3 || SK)$, and checks whether I'_i matches with the received I_i. If it does not hold, GWN terminates the session. Otherwise, the GWN computes $CT_4 = E_{h(IDi \oplus x)}(r_k \oplus r_j || r_i || SID_j || ID_{GWN} || T_2 || T_3 || T_4)$, $V_i = h(SK || T_4 || h(TID_i || x))$, where T_4 is the timestamp. Finally, GWN transmits $\{CT_4, V_i\}$ to U_i.

Step 5: U_i computes $r_k \oplus r_j || r_i || SID_j || ID_{GWN} || T_2 || T_3 || T_4 = D_{h(IDi \oplus x)}(CT_4)$, and checks the timestamp T_4. Then U_i computes $SK = h(r_k \oplus r_i \oplus r_j || T_1 || T_2 || T_3)$, $V'_i = h(SK || T_4 || h(TID_i || x))$, and checks whether V'_i matches with the received V_i. If it holds, U_i completes the authentication. Otherwise, U_i fails to authenticate the GWN.

2.1.4. Password Update Phase

When a user U_i wants to update the password, he/she needs to execute the following steps:

Step 1: U_i inputs ID_i, PW_i into the smart card SC. SC computes $C_i = h(PW_i || b_i)$, $h(TID_i || x) = B_i \oplus C_i$, $h(ID_i || x) = M_i \oplus h(h(ID_i) \oplus C_i)$, $A'_i = h(h(ID_i) || C_i)$, and checks whether A'_i and A_i are equal. If not, SC fails to authenticate U_i, and rejects the request of the password update. Otherwise U_i inputs a new password PW^*_i.

Step 2: SC computes $C^*_i = h_0(PW^*_i || b_i)$, $B^*_i = h(TID_i || x) \oplus C_i \oplus C^*_i$, $M^*_i = h(ID_i || x) \oplus h(h(ID_i) \oplus C^*_i)$ and $A^*_i = h(h(ID_i) || C^*_i)$.

Step 3: Finally, A^*_i,B^*_i, and M^*_i are stored in SC to replace A_i, B_i, and M_i respectively.

2.2. Review of Jung et al.'s Scheme

Jung et al.'s authentication scheme [29] is shown in Figure 3. This scheme consists of three phases: registration, login and authentication, password change. This scheme has not sensor node registration phase. When the sensor node is developed, a shared key K_{GWN_S} between the sensor node and the GWN is assigned.

2.2.1. User Registration

Step 1: A new user U_i selects the identity ID_i and the password PW_i, generates a random number b_i. Then U_i computes $C_i = h(PW_i || b_i)$, U_i transmits $\{ID_i, C_i\}$ to GWN through a secure channel.

Step 2: Upon receipt of the message, the GWN computes $v = h(x_i)$, $N_i = h(ID_i || C_i) \oplus v$, $M_i = h(C_i || v)$. After that, GWN stores $\{v\}$ in its memory, and stores $\{N_i, M_i, h\}$ into smart card SC. Finally, GWN sends SC to U_i via a private channel.

Step 3: After receiving SC from GWN, U_i stores b_i into SC.

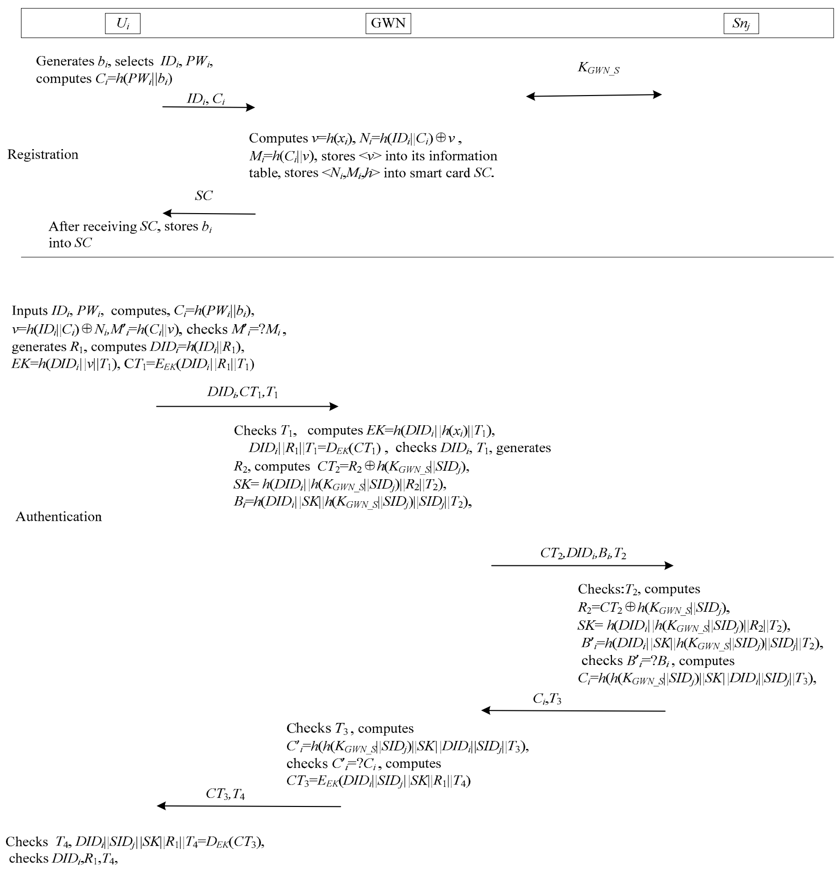

U_i
GWN
Sn_j
Generates b_i, selects ID_i, PW_i, computes C_i=$h(PW_i||b_i)$
K_{GWN_S}
ID_i, C_i
Registration
Computes v=$h(x_i)$, N_i=$h(ID_i||C_i)\oplus v$, M_i=$h(C_i||v)$, stores <v> into its information table, stores <N_i,M_i,h> into smart card SC.
SC
After receiving SC, stores b_i into SC
Inputs ID_i, PW_i, computes, C_i=$h(PW_i||b_i)$, v=$h(ID_i||C_i)\oplus N_i$,M'_i=$h(C_i||v)$, checks M'_i=?M_i, generates R_1, computes DID_i=$h(ID_i||R_1)$, EK=$h(DID_i||v||T_1)$, CT_1=$E_{EK}(DID_i||R_1||T_1)$
DID_i,CT_1,T_1
Checks T_1, computes EK=$h(DID_i||h(x_i)||T_1)$, $DID_i||R_1||T_1$=$D_{EK}(CT_1)$, checks DID_i, T_1, generates R_2, computes CT_2=$R_2\oplus h(K_{GWN\_S}||SID_j)$, SK= $h(DID_i||h(K_{GWN\_S}||SID_j)||R_2||T_2)$, B_i=$h(DID_i||SK||h(K_{GWN\_S}||SID_j)||SID_j||T_2)$,
Authentication
CT_2,DID_i,B_i,T_2
Checks:T_2, computes R_2=$CT_2\oplus h(K_{GWN\_S}||SID_j)$, SK= $h(DID_i||h(K_{GWN\_S}||SID_j)||R_2||T_2)$, B'_i=$h(DID_i||SK||h(K_{GWN\_S}||SID_j)||SID_j||T_2)$, checks B'_i=?B_i, computes C_i=$h(h(K_{GWN\_S}||SID_j)||SK||DID_i||SID_j||T_3)$,
C_i,T_3
Checks T_3, computes C'_i=$h(h(K_{GWN\_S}||SID_j)||SK||DID_i||SID_j||T_3)$, checks C'_i=?C_i, computes CT_3=$E_{EK}(DID_i||SID_j||SK||R_1||T_4)$
CT_3, T_4
Checks T_4, $DID_i||SID_j||SK||R_1||T_4$=$D_{EK}(CT_3)$, checks DID_i,R_1,T_4,

Figure 3. The authentication and key agreement phase of Jung et al.'s scheme.

2.2.2. Login and Authentication

When a user U_i desires the WSNs services, he/she needs to achieve mutual authenticate with GWN and S_{nj}. Figure 3 illustrates the process of mutual authentication for the proposed scheme. In detail, the process is:

Step 1: U_i inputs ID_i and PW_i into the smart card SC. SC computes $C_i = h(PW_i||b_i)$, $v = h(ID_i||C_i) \oplus N_i$, $M'_i = h(C_i||v)$, and compares M'_i with the stored value M_i. If they are not equal, SC terminates the session. Otherwise, SC believes U_i as a legitimate user. Next, SC generates a random number R_1, and computes $DID_i = h(ID_i||R_1)$, $EK = h(DID_i||v||T_1)$, $CT_1 = E_{EK}(DID_i||R_1||T_1)$, where T_1 is the timestamp. Finally, SC sends the login request $\{DID_i, CT_1, T_1\}$ to GWN through the public channel.

Step 2: After receiving the login messages, the GWN first checks the timestamp T_1, and computes $EK = h(DID_i||h(x_i)||T_1)$, $DID_i||R_1||T_1 = D_{EK}(CT_1)$. Then, the GWN checks whether DID_i and T_1 matches with the received values. If they do not hold, GWN terminates the session. Otherwise, the GWN generates a random numnber R_2, and computes $CT_2 = R_2 \oplus h(x_S||SID_j)$, $SK = h(DID_i||h(K_{GWN\_S}||SID_j)||R_2||T_2)$, $B_i = h(DID_i||SK||h(K_{GWN\_S}||SID_j)||SID_j||T_2)$, where T_2 is the timestamp. Finally, the GWN sends $\{CT_2, DID_i, B_i, T_2\}$ to the sensor node S_{nj}.

Step 3: Upon receiving the messages $\{CT_2, DID_i, B_i, T_{2i}\}$ from GWM, S_{nj} first checks the timestamp T_2, and computes $R_2 = CT_2 \oplus h(K_{GWN\_S}||SID_j)$, $SK = h(DID_i||h(K_{GWN\_S}||SID_j)||R_2||T_2)$, $B'_i = h(DID_i||SK||h(K_{GWN\_S}||SID_j)||SID_j||T_2)$. Then, the S_{nj} checks whether B'_i matches with the received B_i. If it does not hold, S_{nj} terminates the session. Otherwise, the S_{nj} computes $C_i = h(h(K_{GWN\_S}||SID_j)||SK||DID_i||SID_j||T_3)$, where T_3 is the timestamp. Finally, S_{nj} transmits $\{C_i, T_3\}$ to GWN.

Step 4: GWN first checks the timestamp T_3, and computes $C'_i = h(h(K_{GWN\_S}||SID_j)||SK||DID_i||SID_j||T_3)$. Then, the GWN checks whether C'_i matches with the received C_i. If it does not hold, GWN terminates the session. Otherwise, the GWN computes $CT_3 = E_{EK}(DID_i||SID_j||SK||R_1||T_4)$, where T_4 is the timestamp. Finally, GWN transmits $\{CT_3, T_4\}$ to U_i.

Step 5: U_i checks the timestamp T_4 and computes $DID_i||SID_j||SK||R_1||T_4 = D_{EK}(CT_3)$. Then U_i checks whether DID_i, R_1, and T_4 matches with the previous values. If it holds, U_i completes the authentication. Otherwise, U_i fails to authenticate the GWN.

2.2.3. Password Update Phase

When a user U_i wants to update the password, he/she needs to execute the following steps:

Step 1: U_i inputs ID_i, PW_i into the smart card SC. SC computes $C_i = h(PW_i||b_i)$, $v = h(ID_i||C_i)\oplus N_i$, $M'_i = h(C_i||v)$, and checks whether M'_i and the stored M_i are equal. If not, SC fails to authenticate U_i, and rejects the request for the password update. Otherwise U_i inputs a new password PW^*_i.

Step 2: SC computes $C^*_i = h_0(PW^*_i||b_i)$, $N^*_i = v\oplus h(ID_i||C^*_i)$, and $M^*_i = h(C^*_i||v)$.

Step 3: Finally, N^*_i and M^*_i are stored in SC to replace N_i and M_i respectively.

2.3. Security Analysis of Two Related Schemes

The security of the above two related schemes will be discussed in this section. Both of them are claimed that they can resist against various kinds of attacks and fulfill the desirable security requirements. However, we find that these two schemes are prone to smart card loss attack. Besides, they also suffer from two design flaws, including the impractical GWN search operation and no provision for perfect forward security.

2.3.1. Smart Card Loss Attack

The smart card loss attack means that the password in the smart card can be guessed offline in the case where the smart card is lost or stolen. The authors of the above two schemes [28,29] have proved that their schemes are secure against this attack. The proofs assume that the identity of the user is unable to be guessed. However, since the identity of the user is a weak strength with low entropy, several articles [32,35,36] have proposed that the identity may be leaked when the smart card is lost or stolen. We now describe the details of this attack.

For Lu et al.'s scheme [28], suppose that the adversary A has obtained the smart card of U_i, and can extract secret information $<A_i,B_i,M_i,b_i,h>$ from it, where $A_i = h(h(ID_i)||C_i)$, $B_i = h(TID_i||x)\oplus C_i$, $M_i = h(ID_i||x)\oplus h(h(ID_i)\oplus C_i)$, $C_i = h(PW_i||b_i)$. Then A can successfully guess the ID_i and PW_i as below.

Step 1: A guesses a candidate pair ID'_i and PW'_i, and computes $C'_i = h(PW'_i||b_i)$, $A'_i = h(h(ID'_i)||C'_i)$.

Step 2: A checks whether A'_i and A_i stored in smart card are equivalent. If they are equal, A can obtain the correct ID_i and PW_i pair. Otherwise, A repeats the steps 1 and 2 until the correct ID_i and PW_i pair is obtained.

For Jung et al.'s scheme [29], the smart card stores $<N_i,M_i,b_i,h>$, where $N_i = h(ID_i||C_i)\oplus v$, $M_i = h(C_i||v)$, $v = h(x_i)$, $C_i = h(PW_i||b_i)$. Therefore, the process of launching smart card loss attack is similar, in many ways, to the process of attacking Lu et al.'s scheme. A can guess the correct ID_i and PW_i pair through checking whether $M_i = h(h(PW'_i||b_i)||N_i\oplus h(ID'_i||h(PW'_i||b_i)))$ holds or not.

Since the identity space $|D_{ID}|$ and the password space $|D_{PW}|$ are usually not more than 10^6, the time required for A to complete this attack is linear [35]. As a result, Lu et al.'s scheme and Jung et al.'s scheme still fail to smart card loss attack.

2.3.2. Impractical GWN Search Operation

User anonymity is an important security feature of authentication scheme for WSNs, which consists of two properties, user identity-protection, and untraceability [36]. User identity-protection means that the adversary could not know the real identity of the user, and user untraceability guarantees that the adversary can neither determine who the user is nor distinguish whether two sessions are executed by the same user. To achieve user anonymity, the 'dynamic ID technique' is widely adopted in most schemes, so do Lu et al.'s scheme [28] and Jung et al.'s scheme [29]. In the two schemes, a user requires concealing his real identity into a dynamic identity. When the user wants to log in GWN, it is difficult for GWN to tell apart the real identity of the user. As a result, the GWN needs to search for every possible parameter or have a back-end channel to figure out the exact user, which is impractical [27]. The detailed of this operation will be described as follows.

For Lu et al.'s scheme, the user sends a login message $\{CT_1,E_i\}$ to GWN, where $CT_1 = E_{EK}(ID_i||T_1||TID_i||r_i)$, $E_i = h(h(ID_i||x)||r_i||T_1)$, $EK = h(TID_i||x) = B_i \oplus C_i$. After receiving the message $\{CT_1,E_i\}$ from the user, the GWN decrypts CT_1 by the symmetric key $EK = h(TID_i||x)$. Now, there is a problem that the GWN does not figure out exactly which TID_i is the communicating user's because all of the users' TID_i are stored in the GWN. The GWN has to perform an exhaustive search operation to obtain the exact user's TID_i. Let L is the size of user's information table, T_h is the execution time for hash operation and T_E is the execution time for the decryption operation. The time complexity of the above operation is $O(L{*}T_h{*}T_E)$. This is obviously impractical.

The similar problem can also be found in Jung et al.'s scheme. The user sends a login message $\{DID_i,CT_1,T_1\}$ to GWN, where $CT_1 = E_{EK}(DID_i||R_1||T_1)$, $EK = h(DID_i||v||T_1)$, $DID_i = h(ID_i||R_1)$, $v = h(x_i)$. After receiving the message $\{DID_i,CT_1,T_1\}$ from the user, GWN decrypts CT_1 by the symmetric key $EK = h(DID_i||h(x_i)||T_1)$, where x_i is the shared symmetric key between the user and the GWN. Since all users' shared symmetric keys are stored in the GWN, the GWN does not figure out exactly which one is the communicating user's. It is obviously unrealistic for the GWN to perform an exhaustive search operation to obtain the exact user's x_i. Because the time complexity of the above operation is $O(L{*}2T_h{*}T_E)$, where L is the size of user's key table, T_h is the execution time for hash operation and T_E is the execution time for the decryption operation.

2.3.3. No Provision for Perfect Forward Secrecy

Perfect forward secrecy is one of the important security properties for authenticated key establishment protocols. A protocol is said to achieve the notion of perfect forward secrecy if the compromise of long-term keys does not compromise the previous session keys [33]. In the practical application, such as battlefield, the sensor node is unattended, which make it be dangerous in compromised by the adversary. Then the long-term key of the sensor node may be compromised and the previous session keys will be retrieved. Therefore, perfect forward secrecy should be considered for WSNs. However, none of the above two schemes [28,29] can provide perfect forward secrecy.

For Lu et al.'s scheme, suppose the user's long-term secret key $h(ID_i||x)$ and $h(TID_i||x)$ are compromised by the adversary A, and A has captured all the previous transmitted messages through the public communication channel. In this case, A is able to obtain all the previous message CT_4. Thus, A can retrieve the past session keys through $r_k \oplus r_j||r_i||SID_j||ID_{GWN}||T_2||T_3||T_4 = D_{h(IDi \oplus x)}(CT_4)$, $SK = h(r_k \oplus r_i \oplus r_j||T_1||T_2||T_3)$. Meanwhile, if the GWN's long-term secret key x and the sensor node's long-term secret key K_{GMN_S} are compromised, the previous session keys will also be retrieved.

The similar problem can also be found in Jung et al.'s scheme, if the user's long-term secret key v is compromised by A, and A has captured all the previous transmitted messages DID_i, CT_3, T_1 through the public communication channel. Thus, A can retrieve the past session keys through $EK = h(DID_i||v||T_1)$, $DID_i||SID_j||SK||R_1||T_4 = D_{EK}(CT_3)$. Meanwhile, if the GWN's long-term secret key x_i and the sensor node's long-term secret key K_{GWN_S} are compromised, the previous session keys will also be retrieved.

3. The Proposed Scheme

This section will describe each phase of the proposed anonymous authentication scheme for WSNs. It uses *PID* instead of the user's real identity to protect user anonymity. In order to achieve the perfect forward secrecy, the transmitted messages in public channel are protected by the one-time hash chain technique. The back-end of GWN stores new *PID* and old *PID* during execution so as to resist against de-synchronization attack. The old *PID* will be set null until the GWN completes the authentication successfully. The proposed scheme consists of four phases: registration phase, authentication and key agreement phase, password update phase, and dynamically deploy sensor nodes phase. We will describe the detail in the upcoming subsection.

3.1. Registration Phase

The registration phase includes user registration phase and sensor node registration. The details of these processes are described as follows. Figure 4 illustrates the registration phase for the proposed scheme.

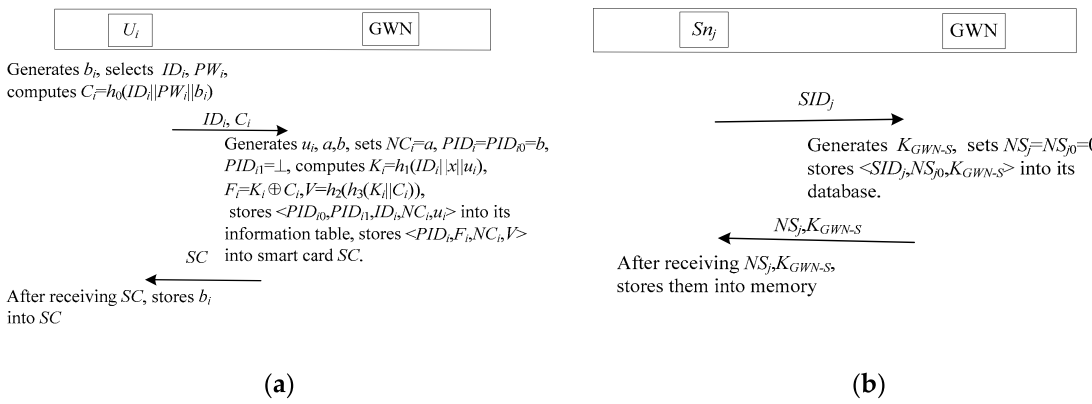

U_i
GWN
Generates b_i, selects ID_i, PW_i, computes C_i=$h_0(ID_i||PW_i||b_i)$
ID_i, C_i
Generates u_i, a,b, sets NC_i=a, PID_i=PID_{i0}=b, PID_{i1}=⊥, computes K_i=$h_1(ID_i||x||u_i)$, F_i=$K_i\oplus C_i$,V=$h_2(h_3(K_i||C_i))$, stores <PID_{i0},PID_{i1},ID_i,NC_i,u_i> into its information table, stores <PID_i,F_i,NC_i,V> into smart card SC.
SC
After receiving SC, stores b_i into SC
Sn_j
GWN
SID_j
Generates $K_{GWN\text{-}S}$, sets NS_j=NS_{j0}=0, stores <SID_j,NS_{j0},$K_{GWN\text{-}S}$> into its database.
NS_j,$K_{GWN\text{-}S}$
After receiving NS_j,$K_{GWN\text{-}S}$, stores them into memory
(**a**)
(**b**)

Figure 4. The registration phase. (**a**) The user registration phase; (**b**) The sensor node registration phase.

3.1.1. User Registration

When a user U_i wants to access a sensor node S_{nj}, he/she needs to register in GWN first. The GWN issues a smart card to U_i as a response to the registration request. As shown in Figure 4a, the procedure of user registration is described as follows.

Step 1: A new user U_i selects identity ID_i and password PW_i, generates a random number b_i. Then U_i computes $C_i = h_0(ID_i||PW_i||b_i)$, U_i transmits $\{ID_i,C_i\}$ to GWN through a secure channel.

Step 2: The GWN checks whether ID_i exists in the user information table. If it exists, GWN rejects the registration request. Otherwise, GWN generates three random numbers u_i,a,b, sets $NC_i = a$, $PID_i = PID_{i0} = b$, $PID_{i1} = \perp$, and computes $K_i = h_1(ID_i||x||u_i)$, $F_i = K_i \oplus C_i$,$V = h_2(h_3(K_i||C_i))$, where $\perp$ denotes null. After that, GWN updates the user identity information table with the new entry $\{PID_{i0}$,PID_{i1},ID_i,NC_i,$u_i\}$, and stores $\{PID_i$,F_i,NC_i,$V\}$ into smart card SC. Finally, GWN sends SC to U_i via a private channel.

Step 3: After receiving SC from GWN, U_i stores b_i into SC.

3.1.2. Sensor Node Registration

When a new sensor node S_{nj} is deployed, S_{nj} is required to register in GWN. As shown in Figure 4b, the procedure of sensor node registration is described as follows.

Step 1: The new sensor node S_{nj} selects identity SID_j and transmits $\{SID_j\}$ to GWN through a secure channel.

Step 2: The GWN checks whether SID_j exists in the sensor node information table. If it exists, the GWN rejects the registration request. Otherwise, the GWN generates a random number $K_{GWN\text{-}S}$, and sets the initial sequence numbers $NS_j = NS_{j0} = 0$. After that, GWN updates the sensor node information table with the new entry $\{SID_j, NS_{j0}, K_{GWN\text{-}S}\}$, and sends $\{NS_j, K_{GWN\text{-}S}\}$ to S_{nj} via a private channel.

Step 3: After receiving $NS_j, K_{GWN\text{-}S}$ from GWN, S_{nj} stores them into its memory as secret.

3.2. Authentication and Key Agreement Phase

When a user U_i wants to gain access to WSNs, U_i needs to achieve mutual authenticate with GWN and S_{nj}. As shown in Figure 5, the process of mutual authentication is described as follows.

Step 1: U_i inputs ID_i and PW_i into the smart card SC. SC computes $C_i = h_0(ID_i || PW_i || b_i)$, $K_i = F_i \oplus C_i$, $V' = h_2(h_3(K_i || C_i))$, and compares V' with the stored value V. If they are not equal, SC terminates the session. Otherwise, SC believes U_i as a legitimate user. Next, SC generates a random number r_A, and computes $EK = h_1(PID_i || K_i || NC_i)$, $CT_1 = E_{EK}(r_A || T)$, $V_1 = h_3(ID_i || r_A || K_i || PID_i || NC_i || T)$, where T is the timestamp. Finally, SC sends the login request $\{PID_i, CT_1, V_1\}$ to GWN through the public channel.

Step 2: After receiving the login messages, GWN at first checks the timestamp T. Then GWN searches its back-end database to get each pair of the pseudonym identity (PID_{i0}, PID_{i1}) and operates as follows:

(1) GWN checks whether the pseudonym identity exists in the user information table.

- If $PID_i = PID_{i0}$, it means that both the user's and GWN's pseudonym identity are updated in the previous session. Then GWN needs to verify whether the one-time hash chain value updates or not. GWN checks whether $PID_{i1} = \perp$ holds.
  - ✧ If the equation does not hold, it means that the GWN's hash chain value does not update in the previous session. So, GWN computes $NC'_i = h_1(NC_i)$, $K_i = h_1(ID_i || x || u_i)$, $EK = h_1(PID_{i0} || K_i || NC'_i)$, $r_A || T = D_{EK}(CT_1)$, $V'_1 = h_3(ID_i || r_A || K_i || PID_{i0} || NC'_i || T)$. The GWN checks whether V'_1 matches with the received V_1. If it holds, GWN generates a random PID'_{i0}, and sets $PID_{i1} = PID_{i0}$, $PID_{i0} = PID'_{i0}$, $NC_i = NC'_i$. Otherwise, GWN terminates the session.
  - ✧ Otherwise, GWN computes $K_i = h_1(ID_i || x || u_i)$, $EK = h_1(PID_{i0} || K_i || NC_i)$, $r_A || T = D_{EK}(CT_1)$, $V'_1 = h_3(ID_i || r_A || K_i || PID_{i0} || NC_i || T)$. The GWN checks whether V'_1 matches with the received V_1. If it holds, GWN generates a random PID'_{i0}, and sets $PID_{i1} = PID_{i0}$, $PID_{i0} = PID'_{i0}$. Otherwise, GWN terminates the session.
- If $PID_i = PID_{i1}$, it means that the user's pseudonym identity and hash chain are not updated in the previous session. GWN computes $K_i = h_1(ID_i || x || u_i)$, $EK = h_1(PID_{i1} || K_i || NC_i)$, $r_A || T = D_{EK}(CT_1)$, $V'_1 = h_3(ID_i || r_A || K_i || PID_{i1} || NC_i || T)$. GWN checks whether V'_1 matches with the received V_1. If it holds, GWN generates a random PID'_{i0}, and sets $PID_{i0} = PID'_{i0}$. Otherwise, GWN terminates the session.
- If $PID_i \neq PID_{i0}$, $PID_i \neq PID_{i1}$, GWN terminates the session.

(2) GWN randomly generates a session key sk and chooses a specified sensor node SID_j, and computes $CT_2 = (sk || ID_i) \oplus h_0(K_{GWN\text{-}S} || SID_j || NS_{j0})$, $V_2 = h_3(ID_i || SID_j || sk || K_{GWN\text{-}S} || NS_{j0})$. Subsequently, GWN updates $K_{GWN\text{-}S} = h_1(K_{GWN\text{-}S} || SID_j)$, $NS_{j0} = NS_{j0} + 1$.

(3) GWN sends $\{CT_2, V_2, NS_{j0}\}$ to the sensor node S_{nj} that U_i wants to interact with via the public channel.

Step 3: Upon receiving the messages $\{CT_2, V_2, NS_{j0}\}$ from GWM, S_{nj} at first verifies whether $1 \leq NS_{j0}\text{-}NS_j \leq N$, where N is a threshold, which sets according to specific requirements of applications. If it does not hold, S_{nj} terminates the session. Otherwise, S_{nj} set $K'_{GWN\text{-}S} = K_{GWN\text{-}S}$, and computes N-1 times $K'_{GWN\text{-}S} = h_1(K'_{GWN\text{-}S} || SID_j)$, if N-1 = 0, there is no hash function operation. Next, S_{nj} computes $sk || ID_i = CT_2 \oplus h_0(K'_{GWN\text{-}S} || SID_j || NS_{j0}\text{-}1)$, $V_2' = h_3(ID_i || SID_j || sk || K'_{GWN\text{-}S} || NS_{j0}\text{-}1)$. Then, S_{nj} checks whether V'_2 matches with the received V_2. If it holds, S_{nj} computes $V_3 = h_3(SID_j || ID_i || sk || NS_{j0})$, and updates $K_{GWN\text{-}S} = h_1(K'_{GWN\text{-}S} || SID_j)$, $NS_j = NS_{j0}$. Otherwise, S_{nj} terminates the session. Finally, S_{nj} transmits $\{SID_j, V_3\}$ to GWN.

Step 4: GWN first computes $V'_3 = h_3(SID_j || ID_i || sk || NS_{j0})$, and checks whether V'_3 matches with the received V_3. If it holds, GWN computes $GEK = h_1(r_A || PID_{i1} || K_i || NC_i)$, $CT_3 = E_{GEK}(sk || PID_{i0} || SID_j)$, $V_4 = h_3(ID_i || sk || r_A || PID_{i0})$. Otherwise, GWN terminates the session. Finally, GWN transmits $\{CT_3, V_4\}$ to U_i.

Step 5: U_i computes $GEK = h_1(r_A || PID_i || K_i || NC_i)$, $sk || PID_{i0} || SID_j = D_{GEK}(CT_3)$, $V_4' = h_3(ID_i || sk || r_A || PID_{i0})$, and checks whether V'_4 matches with the received V_4. If it holds, U_i computes $V_5 = h_3(SID_j || ID_i || PID_{i0} || sk)$, and updates $NC_i = h_1(NC_i)$, $PID_i = PID_{i0}$. Otherwise, U_i terminates the session. Finally, U_i sends $\{V_5\}$ to GWN.

Step 6: After receiving the message V_5 from U_i, GWN computes $V'_5 = h_3(SID_j || ID_i || PID_{i0} || sk)$, and checks whether V'_5 matches with the received V_5. If it holds, GWN updates $NC_i = h_1(NC_i)$, $PID_{i1} = \perp$. Otherwise, GWN fails to authenticate U_i.

Thus, the authentication key agreement among three-party is successful, and they establish the session key sk with each other as summarized in Figure 5.

3.3. Password Update Phase

When a user U_i wants to update the password, he/she needs to run the following steps:

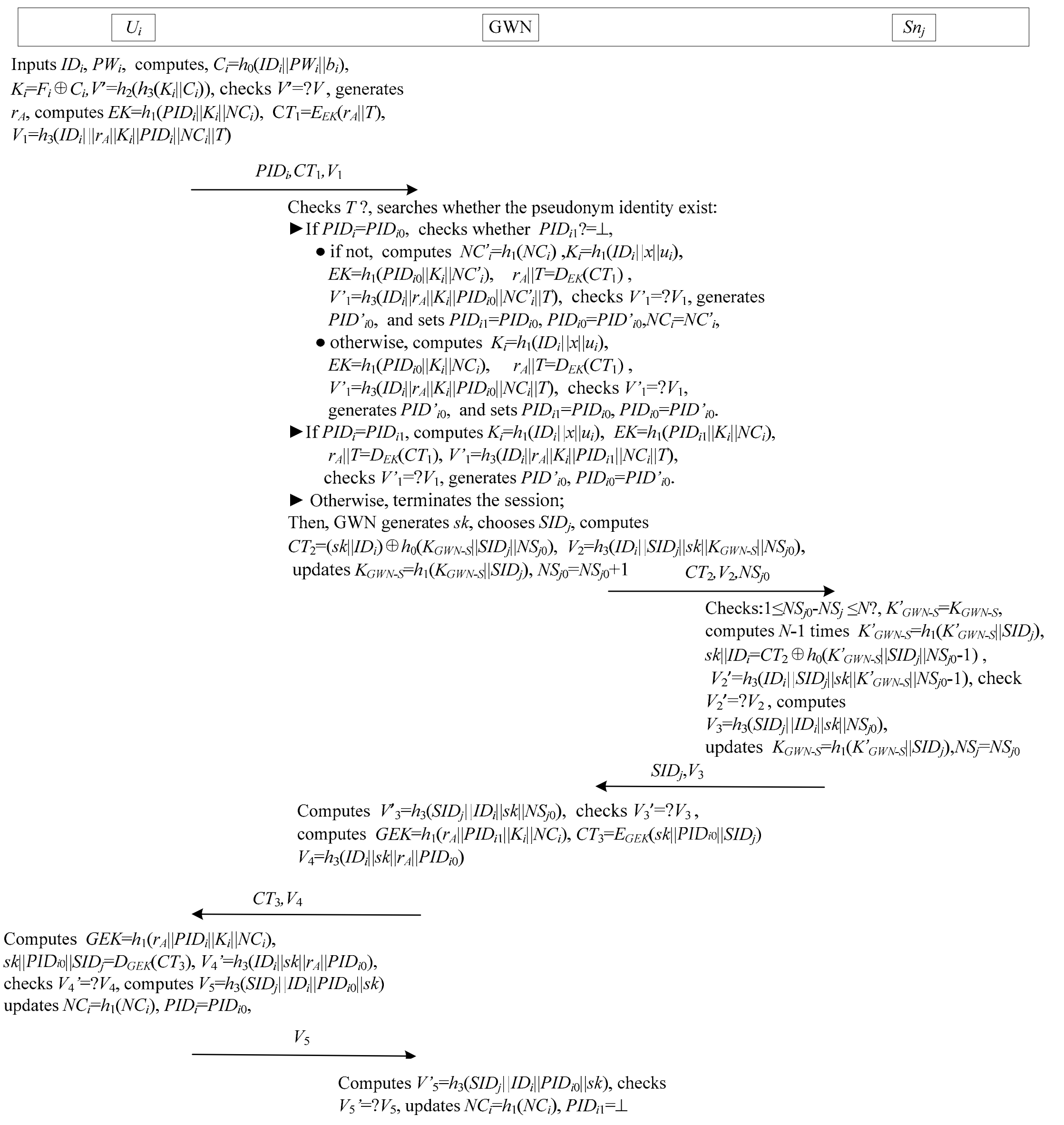
Step 1: U_i inputs ID_i, PW_i into the smart card SC. SC computes $C_i = h_0(ID_i || PW_i || b_i)$, $K_i = F_i \oplus C_i$, $V' = h_2(h_3(K_i || C_i))$, and checks whether V' and V are equal. If not, SC fails to authenticate U_i, and rejects the request of the password update. Otherwise U_i inputs a new password PW^*_i.

Step 2: SC computes $\mathrm{C^*_i = h_0(ID_i || PW^*_i || b_i)}$, $\mathrm{F^*_i = K_i \oplus C_i \oplus C^*_i}$, $\mathrm{V^* = h_2(h_3(K_i || C^*_i))}$.

Step 3: Finally, F^*_i and V^* are stored in SC to replace F_i and V respectively.

3.4. Dynamically Deploy Sensor Nodes Phase

When the system administrator deploys a new sensor node in the existing system, the deployed sensor node is required to apply to register in the GWN. The procedure of sensor node registration follows the steps described in Section 3.1.2.


U_i GWN Sn_j

Inputs ID_i, PW_i, computes, $C_i=h_0(ID_i||PW_i||b_i)$, $K_i=F_i\oplus C_i, V'=h_2(h_3(K_i||C_i))$, checks $V'=?V$, generates r_A, computes $EK=h_1(PID_i||K_i||NC_i)$, $CT_1=E_{EK}(r_A||T)$, $V_1=h_3(ID_i||r_A||K_i||PID_i||NC_i||T)$

$\xrightarrow{PID_i, CT_1, V_1}$

Checks T ?, searches whether the pseudonym identity exist:

► If $PID_i=PID_{i0}$, checks whether $PID_{i1}?=\perp$,

- if not, computes $NC'_i=h_1(NC_i)$,$K_i=h_1(ID_i||x||u_i)$, $EK=h_1(PID_{i0}||K_i||NC'_i)$, $r_A||T=D_{EK}(CT_1)$, $V'_1=h_3(ID_i||r_A||K_i||PID_{i0}||NC'_i||T)$, checks $V'_1=?V_1$, generates PID'_{i0}, and sets $PID_{i1}=PID_{i0}$, $PID_{i0}=PID'_{i0}$,$NC_i=NC'_i$,
- otherwise, computes $K_i=h_1(ID_i||x||u_i)$, $EK=h_1(PID_{i0}||K_i||NC_i)$, $r_A||T=D_{EK}(CT_1)$, $V'_1=h_3(ID_i||r_A||K_i||PID_{i0}||NC_i||T)$, checks $V'_1=?V_1$, generates PID'_{i0}, and sets $PID_{i1}=PID_{i0}$, $PID_{i0}=PID'_{i0}$.

► If $PID_i=PID_{i1}$, computes $K_i=h_1(ID_i||x||u_i)$, $EK=h_1(PID_{i1}||K_i||NC_i)$, $r_A||T=D_{EK}(CT_1)$, $V'_1=h_3(ID_i||r_A||K_i||PID_{i1}||NC_i||T)$, checks $V'_1=?V_1$, generates PID'_{i0}, $PID_{i0}=PID'_{i0}$.

► Otherwise, terminates the session;

Then, GWN generates sk, chooses SID_j, computes $CT_2=(sk||ID_i)\oplus h_0(K_{GWN\text{-}S}||SID_j||NS_{j0})$, $V_2=h_3(ID_i||SID_j||sk||K_{GWN\text{-}S}||NS_{j0})$, updates $K_{GWN\text{-}S}=h_1(K_{GWN\text{-}S}||SID_j)$, $NS_{j0}=NS_{j0}+1$

$\xrightarrow{CT_2, V_2, NS_{j0}}$

Checks:$1\le NS_{j0}\text{-}NS_j\le N$?, $K'_{GWN\text{-}S}=K_{GWN\text{-}S}$, computes N-1 times $K'_{GWN\text{-}S}=h_1(K'_{GWN\text{-}S}||SID_j)$, $sk||ID_i=CT_2\oplus h_0(K'_{GWN\text{-}S}||SID_j||NS_{j0}\text{-}1)$, $V_2'=h_3(ID_i||SID_j||sk||K'_{GWN\text{-}S}||NS_{j0}\text{-}1)$, check $V_2'=?V_2$, computes $V_3=h_3(SID_j||ID_i||sk||NS_{j0})$, updates $K_{GWN\text{-}S}=h_1(K'_{GWN\text{-}S}||SID_j)$,$NS_j=NS_{j0}$

$\xleftarrow{SID_j, V_3}$

Computes $V'_3=h_3(SID_j||ID_i||sk||NS_{j0})$, checks $V_3'=?V_3$, computes $GEK=h_1(r_A||PID_{i1}||K_i||NC_i)$, $CT_3=E_{GEK}(sk||PID_{i0}||SID_j)$ $V_4=h_3(ID_i||sk||r_A||PID_{i0})$

$\xleftarrow{CT_3, V_4}$

Computes $GEK=h_1(r_A||PID_i||K_i||NC_i)$, $sk||PID_{i0}||SID_j=D_{GEK}(CT_3)$, $V_4'=h_3(ID_i||sk||r_A||PID_{i0})$, checks $V_4'=?V_4$, computes $V_5=h_3(SID_j||ID_i||PID_{i0}||sk)$ updates $NC_i=h_1(NC_i)$, $PID_i=PID_{i0}$,

$\xrightarrow{V_5}$

Computes $V'_5=h_3(SID_j||ID_i||PID_{i0}||sk)$, checks $V_5'=?V_5$, updates $NC_i=h_1(NC_i)$, $PID_{i1}=\perp$

Figure 5. The authentication and key agreement phase.

4. Security Analysis of Our Scheme

In the section, we will discuss the security of our proposed scheme. First, the strand space model will be adopted to demonstrate the validity of our scheme. Second, we will demonstrate that our scheme provides mutual authentication and session key security using automated protocol verifier tool ProVerif. Finally, further security analysis illustrates the ability of the proposed scheme to resist various known attacks.

4.1. Authentication Proof Based on Strand Space Model

Strand space model [37,38] is a well-known formal analysis method to verify the security of cryptographic protocols. Before we prove the correctness of our proposed scheme using stand space mode, we will describe the basic notions as below.

4.1.1. The Basic Notion of Strand Space Model

According to [37,38], a stand space is a set Σ of stands with a trace mapping $tr{:}\Sigma \rightarrow (\pm A)^*$, which includes various protocol participant stands and penetrator strands. Where A is a set, the elements of which are the transmitted messages between principals. $(\pm A)^*$ is the set of finite sequences. The elements of A is denoted as terms t. $t_1 \sqsubset t$ is defined as that t_1 is a subterm of t. Due to the limitations of space, only the fundamental notations and lemmas in strand space model are enumerated here:

$+t/-t$: send/receive a term t.
$<s,i>$: a node of s, where $s \in \Sigma$, $1 \leq i \leq$ length($tr(r)$). If n = $<s,i>$, then, index(n) = i, strand(n) = s, term(n) is the ith signed term in the trace of s, and uns_term(n) is the unsigned part of the ith signed term in the strand of s.
$n_1 \rightarrow n_2$: it means that the node n_1 sends a message and n_2 receives the message.
$n_1 \Rightarrow n_2$: it means that n_1 is an immediate causal predecessor of n_2, n_1 = $<s,i>$ and n_2 = $<s,i+1>$.
$n_1 \Rightarrow^+ n_2$: it means that n_1 is a precedence of n_2, n_1 = $<s,i>$ and n_2 = $<s,j>$, $i < j$.
S: a set of edges with respect to the causal relations $\rightarrow$, $\Rightarrow$ and $\Rightarrow^+$.
$n \prec_S n'$: it means that there are one or more edges in S leading from n to n'.
$n \preceq_S n'$: it means that there are zero or more edges in S leading from n to n'.
T: a set of atomic messages.
K: a set of cryptographic keys, which disjoints from T.
$\{m\}_K$: it means that the message m is encrypted by the key K.

Lemma 1. *Suppose C is a bundle. Then $\preceq_C$ is a partial order, i.e., a reflexive, anti-symmetric, transitive relation. Every non-empty subset of the nodes in C has $\preceq_C$-minimal members.*

Lemma 2. *Suppose C is a bundle. and suppose S is a set of nodes such that uns_term(m) = uns_term(m′) implies that m ∈ S iff m′ ∈ S, for all nodes m,m′. If n is a $\preceq_C$-minimal member of S, then the sign of n is positive.*

4.1.2. Penetrator Strands

The following describes the abilities of an adversary, which are mainly characterized by the two factors, the one is the key set K_p possessed by the adversary, the other is the capability of the adversary to generate new messages from messages he intercepts. The strands of the adversary/penetrator are as follows:

M. Text message: $< +t>$, the penetrator sends an atomic messages t, where $t \in T$.
F. Flushing: $<-g>$, the penetrator receives message g.
T. Tee: $<-g, +g, +g>$, the penetrator receives message g and forward it.
C. Concatenation: $<-g,-h, +gh>$, after receiving messages g and h, the penetrator joins them to get gh, then sends gh.
S. Separation into components: $<-gh, +g, +h>$, upon receiving message gh, the penetrator sends message g and h.
K. Key: $< +K>$, the penetrator sends a key K, where $K \in K_P$.
E. Encryption: $<-K,-h, +\{h\}_K>$, after receiving a key K and a message h, the penetrator encrypts h using K, and gets $\{h\}_K$. Then, he sends $\{h\}_K$.
D. Decryption: $<-K^{-1},-\{h\}_K, +h>$, after receiving a private key K^{-1} and a ciphertext $\{h\}_K$, the penetrator decrypts $\{h\}_K$ using K, and gets h. Then, he sends h.
H. Hash:$<-K,-M, +H(K,M)>$, after receiving a key K and a message M, the penetrator compute the hash value of $K||M$, and gets $H\{K||M\}$. Then, he sends $H\{K||M\}$.

4.1.3. Authentication Proof Based on the Stand Space Model

The process of our proposed authentication scheme is as follows:

(1) $U_i \rightarrow$ GWN: PID_i, CT_1, V_1
(2) GWN $\rightarrow S_{nj}$: CT_2, V_2, NS_{j0}
(3) $S_{nj} \rightarrow$ GWN: SID_j, V_3
(4) GWN $\rightarrow U_i$: CT_3, V_4
(5) $U_i \rightarrow$ GWN: V_5

where $EK = h_1(PID_i||K_i||NC_i)$, $CT_1 = E_{EK}(r_A||T)$, $V_1 = h_3(ID_i||r_A||K_i||PID_i||NC_i||T)$, $CT_2 = (sk||ID_i) \oplus h_0(K_{GWN\text{-}S}||SID_j||NS_{j0})$, $V_2 = h_3(ID_i||SID_j||sk||K_{GWN\text{-}S}||NS_{j0})$, $V_3 = h_3(SID_j||ID_i||sk||NS_{j0})$, $GEK = h_1(r_A||PID_{i1}||K_i||NC_i)$, $CT_3 = E_{GEK}(sk||PID_{i0}||SID_j)$, $V_4 = h_3(ID_i||sk||r_A||PID_{i0})$, $V_5 = h_3(SID_j||ID_i||PID_{i0}||sk)$.

The security goal of our proposed scheme is that the three participants should authenticate each other and share a secret key *sk*. In order to make the process of proof description clearer, we will refer to our proposed scheme using abbreviations LAAP.

Definition 1. *Let (Σ,P) be an infiltrated strand space. The strands space model is shown in Figure 6. (Σ,P) is a LAAP space if Σ have four kinds of strands.*

(1) Penetrator strands $s \in P$.
(2) 'User strands' with trace U[ID_i,SID_j,T, r_A,PID_i,PID_{i0},K_i,NC_i,sk], *defined to be* $< +\{PID_i, CT_1, V_1\}, -\{CT_3, V_4\}, +\{V_5\}>$, *where* ID_i, $SID_j \in T_{name}$, r_A, sk,$PID_i \in T$, K_i,$NC_i \in K$, $r_A \notin T_{name}$, $r_A \notin K$.
(3) 'GWN strands' with trace G[ID_i,SID_j,T, r_A,PID_{i0}, PID_{i1},K_i,NC_i,sk,$K_{GWN\text{-}S}$, NS_{j0}],*defined to be* $< -\{PID_i, CT_1, V_1\}, +\{CT_2,V_2,NS_j\}, -\{SID_j, V_3\}, +\{CT_3,V_4\}, -\{V_5\}>$, *where* ID_i, $SID_j \in T_{name}$, r_A, sk,PID_i,PID_{i0},$PID_{i1} \in T$, K_i,NC_i,$K_{GWN\text{-}S} \in K$, $sk \notin T_{name}$, $sk \notin K$.
(4) 'Sensor node' strands with trace Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}],defined to be $<-\{CT_2,V_2,NS_{j0}\}, +\{SID_j, V_3\}>$

If the user, the GWN, and the sensor node can achieve successful authentication with each other, our scheme is a secure authentication scheme. The details in the proof of our proposed scheme using strand space model is described in the Appendix A.

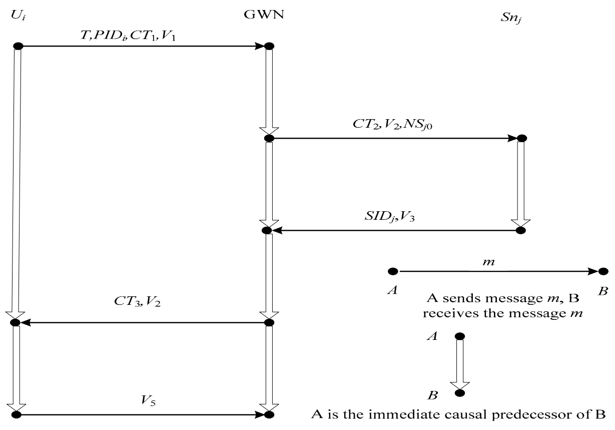

U_i
GWN
Sn_j
T,PID_i,CT_1,V_1
CT_2,V_2,NS_{j0}
SID_j,V_3
m
A
B
A sends message m, B receives the message m
CT_3,V_2
A
B
V_5
A is the immediate causal predecessor of B

Figure 6. LAAP strands space model.

4.2. Formal Security Validation Using ProVerif

In this section, we will demonstrate that our scheme provides mutual authentication and session key security using automated protocol verifier tool ProVerif [39–41]. ProVerif is one of the widely used formal verification tools for cryptography protocols, which supports many cryptographic primitives, including symmetric and asymmetric cryptography, digital signatures, hash functions, Diffie-Hellman key agreements, and signature proofs of knowledge.

In order to analyze the security of our scheme by ProVerif, we define two public channels, c1 is the public channel between the user and the GWN and c2 is the public channel between the GWN and the sensor node. The proposed scheme is modeled as the parallel execution of three distinct processes: the user, the GWN and the sensor node. We have implemented the specifications in the latest version 1.96 of Proverif [42] for three processes. The implementation details of the proposed scheme are provided in the supplementary material available at [43].

ProVerif allows the verifier encrypts some free names using the secrecy session key, and verifies the security of session key by test the secrecy of that free names [41]. As shown in Figure 3, we use four names secretA, secretB, secretC and secretD for secrecy queries to analyze the secrecy of session key *sk*. To verify mutual authentication, we declare eight events:

event beginUGparam(host), event endUGparam(host),
event beginGUparam(host), event endGUparam(host),
event beginGSparam(host), event endGSparam(host),
event beginSGparam(host), event endSGparam(host).

Intuitively, if one participant believes he has completed the scheme with another participant and hence executes the event endXXparam(host), where XX denotes UG, GU, GS, or SG. The results show that our scheme can achieve mutual authentication and session key security. We describe the results of the code as below:

➢ Query not attacker(secretA[]), not attacker(secretB[]),not attacker(secretC[]), not attacker(secretD[])

  ✧ RESULT not attacker(secretA[]), not attacker(secretB[]), not attacker(secretC[]), not attacker(secretD[]) are true.
  ✧ The result means that the adversary has not trace to reconstruct secretA, secretB, secretC, secretD. Hence, the session key *sk* is secure to resist cracking.

➢ Query inj-event(endGUparam(GWN)) => inj-event(beginGUparam(GWN))

  ✧ RESULT inj-event(endGUparam(GWN)) = = > inj-event(beginGUparam(GWN)) is true.
  ✧ This result means that the execution of the event beginGUparam(GWN) is preceded by the execution of the event endGUparam(GWN). Hence, the authentication of the user to the GWN holds.

➢ Query inj-event(endUGparam(user)) => inj-event(beginUGparam(user))

  ✧ RESULT inj-event(endUGparam(user)) = > inj-event(beginUGparam(user)) is true.
  ✧ This result means that the execution of the event beginUGparam(user) is preceded by the execution of the event endUGparam(user). Hence, the authentication of the GWN to the user holds.

➢ Query inj-event(endGSparam(GWN)) => inj-event(beginGSparam(GWN))

  ✧ RESULT inj-event(endGSparam(GWN)) => inj-event(beginGSparam(GWN)) is true.
  ✧ This result means that the execution of the event beginGSparam(GWN) is preceded by the execution of the event endGSparam(GWN). Hence, the authentication of the sensor node to the GWN holds.

- ➢ Query inj-event(endSGparam(SN)) => inj-event(beginSGparam(SN))
  - ✧ RESULT inj-event(endSGparam(SN))=> inj-event(beginSGparam(SN)) is true.
  - ✧ This result means that the execution of the event beginSGparam(GWN) is preceded by the execution of the event endSGparam(GWN). Hence, the authentication of the GWN to the sensor node holds.

4.3. Further Security Analysis of Our Scheme

In this section, the ability of the proposed scheme to resist various known attacks will be analyzed.

4.3.1. Resistance to De-synchronization Attack

Our scheme employs *PID* and one-time hash chain techniques to provide user anonymity and perfect forward secrecy. Hence, it needs an additional synchronization method to maintain the consistency of several one-time values among the user, the GWN, and the sensor node. In the proposed scheme, the consistencies of *PID* and hash chain value will be ensured by using two pseudonym identities $< PID_{i0}, PID_{i1} >$ for the communication between the user and the GWN. For the communication between the GWN and the sensor node, we use the serial number to resist de-synchronization attack. Since the hash function is one way, we let the initiator updates the hash chain value at first. As a result, even if the adversary blocked the message, the hash chain value of the GWN and the sensor node can re-synchronize. In order to make our analysis clearer, a brief framework of our scheme is shown in Figure 7.

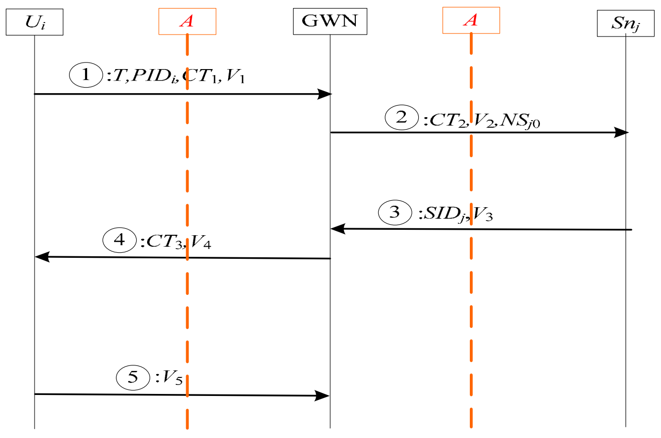


Figure 7. The de-synchronization on our proposed scheme.

The adversary can launch the following malicious scenarios:

Scenario 1: If the adversary blocks the ① message flow, obviously, this attack will not work because all the participants have not even started updating. So, this scenario will be omitted.

Scenario 2: If the ② message flow is blocked by the adversary, the communication will be jammed. For the communication between the U_i and the GWN, this scenario is the same as scenario 4. For the communication between the GWN and S_{nj}, the hash chain values of two participants will not match each other. This attack does not cause our scheme completely unusable because we use serial number NS_{j0} and NS_j to record the number of hash chain updated, where NS_{j0} is the serial number of GWN side, NS_j is the serial number of S_{nj} side. When the GWN sends the ② message flow, the value of hash chain and NS_{j0} in GWN side must be updated. The S_{nj} receives the ② message $\{CT_2, V_2, NS_{j0}\}$, he/she can synchronize the one-time hash chain value through performing NS_{j0}-NS_j time hash functions. Therefore, this scenario will cause asynchronous between the GWN and the S_{nj}, but it will not have any impact on the future session.

Scenario 3: If the adversary blocks the ③ message flow, obviously, this attack will not work between the GWN and the S_{nj} because the two participants have updated their hash chain values, and the hash chain values are equal to each other. For the communication between the U_i and the GWN, this scenario is the same as scenario 4. Therefore, this scenario will be omitted.

Scenario 4: If the ④ message flow is blocked by the adversary, this attack will not work between the GWN and the S_{nj} because both of them have updated hash chain values. But the communication between the U_i and the GWN will be jammed. In this scenario, since both the hash chain values in two participants are not changed, only the synchronization of pseudonym identities are required to consider. The value of PID_{i0} in the GWN side has been a new pseudonym identity, while the value of PID_i in the U_i side does not change. Fortunately, the old pseudonym identity is stored in PID_{i1} in the GWN side, that is $PID_{i1} = PID_i$. So, when the next session is initiated by the U_i using unchanged PID_i, the GWN is still able to recognize it and continues to complete the authentication. Therefore, this scenario will cause pseudonym identity asynchronous between the U_i and the GWN, but it will not have any impact on the future session.

Scenario 5: If the ⑤ message flow is blocked by the adversary, like scenario 4, this attack will not work between the GWN and the S_{nj}. However, for the communication between the U_i and the GWN, it will be jammed. Since the pseudonym identities values of two participants have updated, it means $PID_{i0} = PID_i$, we only need to worry about the synchronization of two participants' hash chain values. In this scenario, the hash chain value in the U_i side is updated, while the value hash chain in the GWN side is unchanged. When U_i using changed hash chain value initiates a new session, the GWN will update its hash chain value through checking whether the value of PID_{i1} is non-null or not. Therefore, even if this scenario will cause hash chain value asynchronous between the U_i and the GWN, the two pseudonym identities will make the hash chain values synchronize again. As a result, our scheme can resist de-synchronization attack through the above analyses.

4.3.2. Mutual Authentication

According to the proofs of Proposition A1–Proposition A4 and the formal validation using ProVerif, it is infeasible for an adversary to forge a legitimate user's or GWN's or sensor node's authentication message. Thus, the user, the GWN, and the sensor node can successfully authenticate each other.

4.3.3. User Anonymity

To protect user's identity, the proposed scheme employs pseudonym identity as a transmitted message instead of user's real identity. The pseudonym identity is randomly generated and changes after completing each session. Thus, the pseudonym identity is different for every session. Moreover, it is almost impossible for an adversary to get the user's real identity from transmitted messages. Therefore, our scheme is able to support user anonymity and untraceability.

4.3.4. Perfect Forward Secrecy

In the proposed scheme, suppose the adversary has obtained the long-term keys of two participants, that are K_i, NC_i, and $K_{GWN\text{-}S}$, he/she still cannot get the session key *sk*. The reason is that after each successful session, the keys NC_i and $K_{GWN\text{-}S}$ will be updated by one-way hash function, that is $NC'_i = h_1(NC_i)$, $K'_{GWN\text{-}S} = h_1(K_{GWN\text{-}S} || SID_j)$. Because the hash function is one way, the adversary cannot obtain NC_i and $K_{GWN\text{-}S}$ from NC'_i and $K'_{GWN\text{-}S}$. Therefore, our scheme can provide perfect forward secrecy.

4.3.5. Resistance to Smart Card Loss Attack

Suppose the adversary steals the user's smart card and obtains the data $\{PID_i, F_i, NC_i, V, b_i\}$, where $K_i = h_1(ID_i || x || u_i)$, $F_i = K_i \oplus C_i, V = h_2(h_3(K_i || C_i))$, $C_i = h_0(ID_i || PW_i || b_i)$. The adversary cannot guess the correct password, because there exist $|\mathrm{D_{ID}}| * |\mathrm{D_{PW}}| / 1024$ candidates of the password, where $|D_{ID}|$ is the space of the identity and $|D_{PW}|$ is the space of the password. This method is called 'fuzzy verifier' [23,44,45], which prevents the adversary from obtaining the exacting correct password. Therefore, our proposed scheme can resist smart card loss attack.

4.3.6. Resistance Known Session-Specific Temporary Information Attack

In the proposed scheme, suppose the adversary gets the ephemeral random number r_A, he still cannot obtain information of session key *sk*. The reason is that the adversary has no way to compute the long-term key K_i, one-time hash chain values NC_i and $K_{GWN\text{-}S}$. Moreover, transmitted messages in the public channel are unhelpful to compute *sk*. Therefore, the proposed scheme has the ability to prevent the session-specific temporary information attack.

4.3.7. Resistance to Stolen Verifier Table Attack

In our scheme, no any password-verifier table of the user is stored in the GWN side. Therefore, our scheme can resist stolen verifier table attack.

4.3.8. Resistance to User Impersonation Attack

In our scheme, in order to forge a user, the adversary has to generate a valid value $\{T, PID_i, CT_1, V_1\}$. However, it is infeasible because the adversary does not know the secret keys K_i and NC_i. Therefore, our proposed scheme can resist against user impersonation attack.

4.3.9. Resistance to Sensor Node Spoofing Attack

Proposition A1–Proposition A4 and the formal validation using ProVerif show that the adversary cannot forge a legitimate user's or sensor node's authentication message without the secret keys K_i, NC_i or K_{GWN_S}. In the proposed scheme, the sensor node only has his own secret value and does not know the secret values of other sensor nodes or users. Therefore, he cannot spoof any user or other sensor nodes.

4.3.10. Resistance to Replay Attack

The proposed scheme uses timestamp, nonce and serial number to prevent the replay attack. For the communication between the user and the GWN, the first message flow includes a current timestamp T, and other message flow employs challenge-response mechanism to resist reply attack. For the communication between the GWN and the sensor node, the serial number is used in every message flow, which is updated after each successful authentication session. As a result, when the user and the sensor node accept each other, it must be the current session, not previous session. Therefore, our proposed scheme can avoid the replay attack.

4.3.11. Resistance to Man-in-the-middle Attack

In the proposed scheme, the transmitted messages are protected by the secret values K_i, NC_i and K_{GWN_S}, anyone without them cannot forge legal authentication messages. Therefore, our scheme can resist man-in-the-middle attack.

4.3.12. Resistance to Wrong Password Login/Update Attack

In the proposed scheme, the password verification information $V = h_2(h_3(K_i || C_i))$ is stored in the mobile device, which is designed to check the correctness of password. If the user inputs wrong password PW'_i, the verification data V and $V' = h_2(h_3(F_i \oplus h_0(ID_i || PW'_i || b_i) || h_0(ID_i || PW'_i || b_i)))$ will not be equal. Therefore, our scheme can quickly detect unauthorized login and password update.

4.4. Security Comparisons

The security features of our proposed scheme with the two prior related schemes [28,29] will be compared in this section. The results of the comparison are listed in Table 2.

Table 2. Security features comparisons of our scheme and the two related schemes.

Features	Lu et al. [28]	Jung et al. [29]	Ours
Resistance to de-synchronization attack	√	√	√
Mutual authentication	√	√	√
User anonymity	√	√	√
Perfect forward security	×	×	√
Smart card loss attack	×	×	√
Resistance to known session-specific temporary information attack	√	√	√
Resistance to stolen verifier table attack	√	√	√
Resistance to user impersonation attack	√	√	√
Resistance to sensor node spoofing attack	√	√	√
Resistance to replay attack	√	√	√
Resistance to man-in-the-middle attack	√	√	√
Resistance to wrong password login/update attack	√	√	√

From Table 2, it can be concluded that the proposed scheme is the only one who can resist against various kinds of known attacks and fulfill the desirable security features. Therefore, our scheme has better security than the previously related schemes.

5. Performance Analysis

This section will compare the communication and communication costs of our proposed scheme with the two prior related schemes [28,29]. Since the registration phase and password update phase are not used frequently, we only concentrate on comparing authentication phase.

5.1. Computation Analysis

For efficiency analysis, we compare the computation costs of our scheme with the two prior related schemes [28,29]. To facilitate analysis, we use the following notations to measure computation costs.

- T_h: the time complexity of the general hash function.
- $T_{E/D}$: the time complexity of general symmetric-key encryption/decryption algorithm.

As pointed out in [46,47], the running time of a one-way hash function operation, and symmetric-key encryption/decryption operation are 0.00032s and 0.0056s respectively. Thus, we have $T_h \approx 0.00032$s, $T_{E/D} \approx 0.0056$s. The results of the computation complexity comparisons of our scheme and two related schemes are summarized in Table 3. It shows that our scheme is as efficient as the most efficient one of these prior related schemes at sensor nodes. Although the computation cost for the user and the GWN of our proposed scheme is higher than that of Jung et al.'s scheme, it should be toleratable because our proposed scheme provides higher security, and resists most well-known attacks.

Table 3. Computation complexity comparisons of our scheme and the two related schemes.

Schemes	Users	GWN	Sensor Node	Total
Lu et al. [28]	$7T_h + 2T_{E/D} \approx 0.01344$s	$8T_h + 4T_{E/D} \approx 0.02496$s	$4T_h + 2T_{E/D} \approx 0.01248$s	$19T_h + 8T_{E/D} \approx 0.05088$s
Jung et al. [29]	$5T_h + 2T_{E/D} \approx 0.0128$s	$5T_h + 2T_{E/D} \approx 0.0128$s	$4T_h \approx 0.00128$s	$13T_h + 4T_{E/D} \approx 0.02688$s
Ours	$9T_h + 2T_{E/D} \approx 0.01408$s	$11T_h + 2T_{E/D} \approx 0.01472$s	$4T_h \approx 0.00128$s	$25T_h + 4T_{E/D} \approx 0.03008$s

5.2. Communication Analysis

In this section, we compare the communication cost of our proposed scheme with the two prior related schemes [28,29]. To achieve convincing comparisons, we assume that the bit length of identity (ID_i,SID_j,ID_{GWN}), password (PW_i), pseudonym identity (PID_i,PID_{i0},PID_{i1}), timestamp (T,T_1,T_2,T_3,T_4), serial number (NS_{j0}, NS_j), random number (r_A,sk), hash (h,h_1,h_3) output and hash (h_0) output are 64, 64, 64, 160, 64, 256, 160 and 320 bits, the block length of the symmetric encryption is 128 bits, respectively. Since the bit length of ciphertext using the symmetric encryption is the multiples of 128 bits, the bit length of CT_1 and CT_3 are 512 and 384 bits, respectively. Table 4 shows the communication cost comparison among our scheme and the prior related schemes. In our scheme, the message $\{PID_i$,CT_1,$V_1\}$, $\{CT_2$,V_2,$NS_{j0}\}$, $\{SID_j$,$V_3\}$, $\{CT_3$,$V_4\}$ and $\{V_5\}$ require (64 + 512 + 160) = 736, (320 + 160 + 64) = 544, (160 + 64) = 224, (384 + 160) = 544 and 160 bits, respectively. Adding the five values, the total communication cost of our scheme is 2208 bits.

For Lu et al.'s scheme [28], the message $\{CT_1$,$E_i\}$, $\{CT_2$,$C_i\}$, $\{CT_3$,$I_i\}$ and $\{CT_4$,$V_i\}$require ((64 + 160 + 64 + 256) + 160) = 128 $\times$ 5 + 160 = 800, ((256 + 64 + 160 + 160) + 160) = 128 $\times$ 5 + 160 = 800, ((256 + 256 + 160) + 160) = 128 $\times$ 6 + 160 = 928, and ((256 + 256 + 64 + 64 + 160 $\times$ 3) + 160) = 128 $\times$ 9 + 160 = 1312 bits, respectively. Adding the four values, the total communication cost of Lu et al.'s scheme is 3840 bits.

For Jung et al.'s scheme [29], the message $\{DID_i$,CT_1,$T_1\}$, $\{CT_2$,DID_i,B_i,$T_2\}$, $\{C_i$,$T_3\}$ and $\{CT_3$,$T_4\}$ require (64 + (64 + 256 + 160) + 160) = 64 + 128 $\times$ 4 + 160 = 736, (256 + 64 + 160 + 160) = 640, (160 + 160) = 320, and ((64 + 64 + 160 + 256 + 160) + 160) = 128 $\times$ 6 + 160 = 928 bits, respectively. Adding the four values, the total communication cost of Jung et al.'s scheme is 2624 bits.

Using the above similar approach, the total communication cost of the other related schemes can be computed in Table 4. From comparison in Table 4, it can be concluded that the proposed scheme has the least communication cost among the above schemes.

Table 4. Communication cost comparisons of our scheme and the two related schemes.

Schemes	Number of Message Required	Number of Bits Required
Lu et al. [28]	4 Messages	3840
Jung et al. [29]	4 Messages	2624
Ours	5 Messages	2208

6. Conclusions

In this paper, we propose a lightweight anonymous authentication protocol for WSNs based on a one-time hash chain and pseudonym identity. The proposed scheme can provide mutual authentication, user anonymity, perfect forward secrecy, etc. Besides, it is resilient to various kinds of known attacks, such as de-synchronization attack, and known session-specific temporary information attack. Formal security analysis and simulations are also conducted by ProVerif to demonstrate that our scheme is secure against active and passive attacks. Furthermore, the proposed scheme only uses symmetric key encryption/decryption and hash functions. It is very suitable for the resource constrained sensor nodes.

Acknowledgments: This research was supported by National Science Foundation of China under Grant number (Grant No. 61502034 and 61571375) and the open research fund of key laboratory of intelligent network information processing, Xihua University (16224221).

Author Contributions: Conceived and designed the experiments: Ling Xiong and Daiyuan Peng. Performed the experiments: Tu Peng and Zhicai Liu. Analyzed the data: Ling Xiong and Daiyuan Peng. Contributed reagents/materials/analysis tools: Zhicai Liu and Hongbin Liang. Wrote the paper: Ling Xiong and Hongbin Liang.

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

Appendix A. The Details in the Proof of Our Proposed Scheme Using Strand Space Model

Proposition A1.
Suppose:

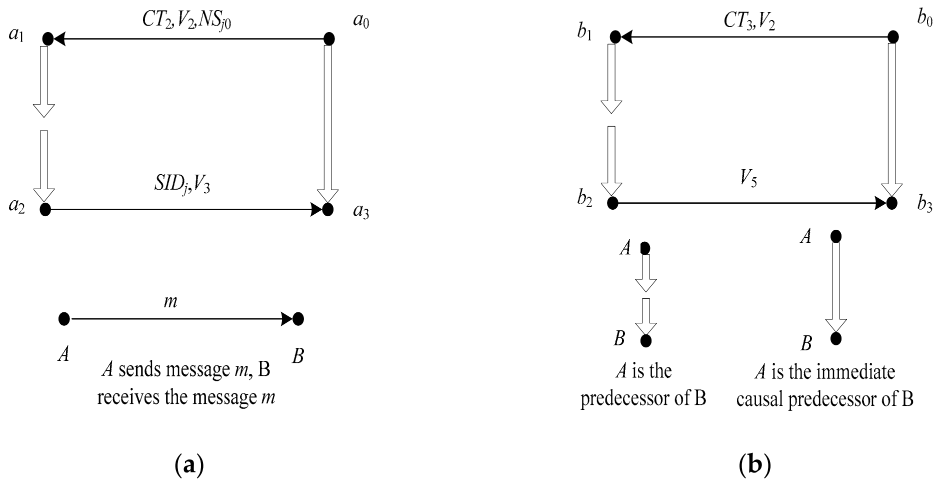
(1) Σ is a LAAP space and C is a bundle containing a GWN's strand s with trace G[ID_i,SID_j,T, r_A,PID_{i0}, PID_{i1},K_i,NC_i,sk,$K_{GWN\text{-}S}$, NS_{j0}];

(2) $EK \notin K_P$, $K_{GWN\text{-}S} \notin K_P$, where $EK = h_1(PID_i \mid\mid K_i \mid\mid NC_i)$; and

(3) $r_A \neq PID_{i0} \neq sk \neq PID_i \neq PID_{i1}$, PID_{i0} and sk are uniquely originating in Σ.

then C contains a user's strand with trace U[ID_i,SID_j,T,r_A,PID_i,PID_{i0},K_i,NC_i,sk], and a sensor node's strand with Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}].

We will prove Proposition A1 using the following lemmas. For the sake of convenience, we will refer to <s,2> (that is the second node + {CT_2,V_2,NS_j} of s) as a_0, and to its term(that is term(a_0)) as u_0. We will refer to <s,3> (that is the third node + {SID_j, V_3} of s) as a_3, and to its term(that is term(a_3)) as u_3. We will refer to other nodes similarly. The node <s,4> is denoted as b_0, and term(b_0) = v_0. The node <s,5> is denoted as b_3, and term(b_3) = v_3. As shown in Figure A1, we will use four additional nodes a_1,a_2,b_1,b_2 during the course of the proof, such that $a_0 \prec a_1 \prec a_2 \prec a_3$, $b_0 \prec b_1 \prec b_2 \prec b_3$.



CT_2,V_2,NS_{j0}
a_1
a_0
SID_j,V_3
a_2
a_3
m
A
B
A sends message m, B receives the message m
(a)
CT_3,V_2
b_1
b_0
V_5
b_2
b_3
A
B
A is the predecessor of B
A
B
A is the immediate causal predecessor of B
(b)

Figure A1. The nodes of a GWN's strand. (**a**) The nodes $a_0 \prec a_1 \prec a_2 \prec a_3$; (**b**) The nodes $b_0 \prec b_1 \prec b_2 \prec b_3$.

First, we will prove C contains a sensor node's strand with Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}].

Lemma A1. *sk originates at a_0.*

Proof. By the assumptions, $sk \sqsubset u_0$, and the sign of a_0 is positive. According to the definition of originates, we only require verifying that sk is not the subterm of a node a', where a' is the precedence node of a_0. Since on the same strand, the precedence node of a_0 is <s,1>, uns_term(<s,1>) = -{PID_i,CT_1,V_1}, where $EK = h_1(PID_i \mid\mid K_i \mid\mid NC_i)$, $CT_1 = E_{EK}(r_A \mid\mid T)$, $V_1 = h_3(ID_i \mid\mid r_A \mid\mid K_i \mid\mid PID_i \mid\mid NC_i \mid\mid T)$. We need to check that $sk \neq r_A$, $sk \neq PID_i$, which is a hypothesis, $sk \neq K_i$, $sk \neq NC_i$, $sk \neq SID_j$, $sk \neq ID_i$, which follows from the stipulation in Definition 1 Clause 3 that $sk \notin T_{name}$, $sk \notin K$. □

Lemma A2. *The set $S = \{a \in C\colon sk \sqsubset term(a) \wedge u_0! \sqsubset term(a)\}$ has a minimal node a_2, $u_0! \sqsubset term(a)$ denotes as u_0 that is the subterm of term(a). The node a_2 is regular and the sign of a_2 is positive.*

Proof. Because $a_3 \in C$, $sk \in C$, $u_0! \sqsubset u_3$, S is non-empty. Therefore, S has at least a minimal element a_2 by Lemma 1, and the sign of the node a_2 is positive by Lemma 2. Whether a_2 lie on a penetrator stand p? We will check it through the form of the trace of p. □

M. Text message: < + t>, where $t \in T$. If this stand contains the node a_2, which means that sk originates on this strand. Accord to Lemma A1, sk originates at a_0. Obviously, it is a contradictory assumption. Therefore, M strand cannot contain the node a_2.

F. Flushing: <-g>, since this strand lacks any positive nodes, F strand cannot contain the node a_2.

T. Tee: <-g, + g, + g>, the penetrator receives message g and forward it. T strand cannot contain the node a_2 because a_2 is a minimal element.

C. Concatenation: <-g,-h, + gh>, after receiving messages g and h, the penetrator joins them to get gh, then sends gh. C strand cannot contain the node a_2 because a_2 is a minimal element.

K. Key: < + K>, the penetrator sends a key K,where $K \in K_P$. K strand cannot contain the node a_2 because $sk! \sqsubset K$.

E. Encryption: <-K,-h, + $\{h\}_K$>, after receiving a key K and a message h, the penetrator encrypts h using K, and gets $\{h\}_K$. Then, he sends $\{h\}_K$. Suppose the node + $\{h\}_K \in S$, since $sk \sqsubset \{h\}_K \wedge u_0! \sqsubset \{h\}_K$, thus $sk \sqsubset h, \wedge u_0! \sqsubset h$. E strand cannot contain the node a_2 because the positive node is not in S.

D. Decryption: <-K^{-1},-$\{h\}_K$, + h>, after receiving a private key K^{-1} and a ciphertext $\{h\}_K$, the penetrator decrypts $\{h\}_K$ using K, and gets h. Then, he sends h. If the node + h is a minimal element a_2, then $u_0! \sqsubset h \wedge u_0 \sqsubset \{h\}_K$. According to the assumption of free encryption, $h = sk||ID_i$, and $K = h_0(K_{GWN\text{-}S}||SID_j||NS_{j0})$. So, there must exist a penetrator strand which can send K. Obviously, it is contradictory because $K_{GWN\text{-}S} \notin K_P$. Therefore, D strand cannot contain the node a_2.

H. Hash:<-K,-M, + $H(K,M)$>, after receiving a key K and a message M, the penetrator compute the hash value of $K||M$, and gets $H\{K||M\}$. Then, he sends $H\{K||M\}$. According to the assumption, $K = sk$, and $M = SID_j||ID_i||NS_{j0}$. So, there must exist a penetrator strand which can send K. Obviously, it is contradictory because sk is secret, and any penetrator without $K_{GWN\text{-}S}$ can know it. Therefore, H strand cannot contain the node a_2.

S. Separation into components: <$-gh$, + g, + h>, upon receiving message gh, the penetrator sends message g and h. Suppose, the unsigned term minimal element uns_term(a_2) = g(there is a similar case if uns_term(a_2) = h). Since $a_2 \in S$, then $sk \sqsubset g \wedge u_0! \sqsubset g$. There must have $u_0 \sqsubset gh$, or the term of the minimal element in S is gh. Because, $u_0! \sqsubset g$, $u_0 \sqsubset gh$, there must have $u_0 \sqsubset h$. Let a set $T = \{m \in C\colon m \prec a_2 \wedge gh \sqsubset \text{term}(m)\}$. Every element of T is a penetrator node because there is no regular node contains a subterm gh, where $u_0 \sqsubset h$. T is non-empty because of the first node of S strand -$gh \in C$. Therefore, T has at least a minimal element m by Lemma 1, and the sign of the node m is positive by Lemma 2. We will check what kind of stand m can lie on.

Obviously, similar to the above analysis, M,F,T,C,K,S,E,D cannot contains the minimal element of T. Therefore, the node a_2 does not lie on p but must lie on a regular node.

Lemma A3. *The precedence node of a_2 is on the same regular strand. Let it as a_1, and term(a_1) = $\{CT_2,V_2,NS_j\}$.*

Proof. Because sk originates at a_0 and sk are uniquely originating in Σ, sk must not originate at a_2. According to Lemma A2, a_2 lie on a regular node, there must be a node preceding a_1 on the same strand such that $sk \sqsubset \text{term}(a_1)$. Since $u_0! \sqsubset \text{term}(a_2)$, and a_2 is the minimal element, $u_0 \sqsubset \text{term}(a_1)$. Besides, there is no regular node contains u_0. Therefore, term(a_1) = $\{CT_2,V_2,NS_j\}$. □

Lemma A4. *The strand containing a_1 and a_2 is a sensor node strand, and is contained in C.*

Proof. According to Lemma A3, term(a_1) = $\{CT_2,V_2,NS_j\}$, and it is the precedence node of a_2. Because a_2 is the minimal element, and it is a positive node. Beside, a_0 is a positive node, and term(a_0) = $\{CT_2,V_2,NS_j\}$. Therefor, a_1 = -$\{CT_2,V_2,NS_j\}$, a_2 = + $\{SID_j,V_3\}$, it must be an sensor node strand. □

Lemma A3 and Lemma A4 shows C contains a sensor node's strand with Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}]. Moreover, we can prove that C contains a user node's strand with U[ID_i,SID_j,r_A,PID_i,PID_{i0},K_i,NC_i,sk] using the above similar proof methods.

Proposition A2. *If Σ is an LAAP space, C is a bundle, and sk are uniquely originating in Σ, then there are at most one sensor node strand t_1 with trace Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}] for any GWN, sensor node and sk.*

Proof. If any sensor node strand t_1 has trace Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}] any user, GWN, and sk, the $<t_1,2>$ is positive, $sk \sqsubset$ term$<t_1,2>$, and sk is the challenge information of GWN. Hence, if sk originates uniquely in Σ, there can be at most one such t_1. □

Proposition A3. *If Σ is an LAAP space, C is a bundle, and r_A is uniquely originating in Σ, then there are at most one user strand t_2 a with trace U[ID_i,SID_j,r_A,PID_i,PID_{i0},K_i,NC_i,sk] for any user, GWN, and sk.*

Proof. If any user strand t_2 has trace U[ID_i,SID_j,r_A,PID_i,PID_{i0},K_i,NC_i,sk] any user, GWN, and sk, the $<t_2,1>$ is positive, $r_A \sqsubset$ term$<t_2$, 1>, and r_A cannot possibly occur earlier on t_2. Therefore, r_A originates at node $<t_2$, 1>. Hence, if r_A originates uniquely in Σ, there can be at most one such t_2. □

Proposition A4.

Suppose:

(1) Σ is a LAAP space and C is a bundle containing a user's strand s with trace U[ID_i,SID_j,r_A,PID_i,PID_{i0},K_i,NC_i,sk] and a sensor node's strand t with Sn[ID_i,SID_j,sk,$K_{GWN\text{-}S}$,NS_{j0}];

(2) $GEK \notin K_P$, $K_{GWN\text{-}S} \notin K_P$, where $GEK = h_1(r_A || PID_{i1} || K_i || NC_i)$; and

(3) $r_A \neq PID_{i0} \neq sk \neq PID_i \neq PID_{i1}$, PID_{i0} and sk are uniquely originating in Σ.

then C contains a GWN's strand with trace G[ID_i,SID_j,r_A,PID_{i0}, PID_{i1},K_i,NC_i,sk,$K_{GWN\text{-}S}$, NS_{j0}].

Proof. In order to prove Proposition A4, we can refer to the GWN's strand as a responder's strand of the user or a initiator's stand of the sensor node. As the responder's strand and as the initiator's stand, the processes of proofs are similar with Proposition A1. Take the case of GWN as an responder's strand, we only consider the set $S = \{a \in C$: $\{V_4\} \sqsubset$ term$(a)\}$ is non-empty. Because $<s_2$,2> is the element of S, S has at least a minimal element a_0. If a_0 is on the regular strand g, it is easy to prove $g \in$ G[ID_i,SID_j,r_A,PID_{i0}, PID_{i1},K_i,NC_i,sk,$K_{GWN\text{-}S}$, NS_{j0}]. Otherwise, g should be on the H strand with trace$<-GEK,-r_A || sk || PID_0$, + $\{V_4\}>$. Obviously, it is contradictory because $GEK \notin K_P$. Therefore, C contains a GWN's strand with trace G[ID_i,SID_j,r_A,PID_{i0}, PID_{i1},K_i,NC_i,sk,$K_{GWN\text{-}S}$, NS_{j0}]. Proposition A1–Proposition A4 show that our scheme is a secure mutual authentication scheme among the user, the GWN and the sensor node. □

References

1. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless sensor network survey. *Comput. Netw.* **2008**, *52*, 2292–2330. [CrossRef]
2. O'Donovan, T.; O'Donoghue, J.; Sreenan, C.; Sammon, D.; O'Reilly, P.; O'Connor, K.A. A Context Aware Wireless Body Area Network (BAN). In Proceedings of the 3rd International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth 2009), London, UK, 1–3 April 2009.
3. Hart, J.K.; Martinez, K. Environmental Sensor Networks: A revolution in the earth system science? *Earth Sci. Rev.* **2006**, *78*, 177–191. [CrossRef]
4. Tiwari, A.; Ballal, P.; Lewis, F.L. Energy-efficient wireless sensor network design and implementation for condition-based maintenance. *ACM Trans. Sens. Netw.* **2007**, *3*, 1. [CrossRef]

5. Gnawali, O.; Jang, K.Y.; Paek, J.; Vieira, M.; Govindan, R.; Greenstein, B.; Joki, A.; Estrin, D.; Kohler, E. The tenet architecture for tiered sensor networks. In Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, 31 October–3 November 2006; ACM: New York, NY, USA, 2006; pp. 153–166. [CrossRef]
6. Yang, D.; Misra, S.; Fang, X.; Xue, G.; Zhang, J. Two-tiered constrained relay node placement in wireless sensor networks: Computational complexity and efficient approximations. *IEEE Trans. Mob. Comput.* **2012**, *11*, 1399–1411. [CrossRef]
7. He, D.; Kumar, N.; Chilamkurti, N. A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks. *Inf. Sci.* **2015**, *321*, 263–277. [CrossRef]
8. He, D.; Chen, C.; Chan, S.; Bu, J.; Yang, L.L. Security Analysis and Improvement of a Secure and Distributed Reprogramming Protocol for Wireless Sensor Networks. *IEEE Trans. Ind. Electron.* **2013**, *60*, 5348–5354. [CrossRef]
9. Perrig, A.; Stankovic, J.; Wagner, D. Security in wireless sensor networks. *Commun. ACM* **2004**, *47*, 53–57. [CrossRef]
10. Koyama, K.; Maurer, U.M.; Okamoto, T.; Vanstone, S.A. New public-key schemes based on elliptic curves over the ring Zn. In *Advances in Cryptology-CRYPTO'91*; Springer: Berlin/Heidelberg, Germany, 1992; pp. 252–266.
11. Watro, R.; Kong, D.; Cuti, S.F.; 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 October 2004; pp. 59–64.
12. Hwang, M.S.; Li, L.H. A new remote user authentication scheme using smart cards. *IEEE Trans. Consum. Electron.* **2000**, *46*, 28–30. [CrossRef]
13. Wong, K.H.; Zheng, Y.; Cao, J.; Wang, S. A dynamic user authentication scheme for wireless sensor networks. In Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, Taichung, Taiwan, 5–7 June 2006; pp. 1–9.
14. Das, M.L. Two-Factor user authentication in wireless sensor networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1086–1090. [CrossRef]
15. Huang, H.F.; Chang, Y.F.; Liu, C.H. Enhancement of two-factor user authentication in wireless sensor networks. In Proceedings of the 2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Darmstadt, Germany, 15–17 October 2010; pp. 27–30.
16. Chen, T.H.; Shih, W.K. A robust mutual authentication protocol for wireless sensor networks. *ETRI J.* **2010**, *32*, 704–712. [CrossRef]
17. Khan, M.K.; Alghathbar, K. Cryptanalysis and security improvements of two-factor user authentication in wireless sensor networks. *Sensors* **2010**, *10*, 2450–2459. [CrossRef] [PubMed]
18. Jiang, Q.; Ma, Z.; Ma, J.; Li, G. Security enhancement of robust user authentication framework for wireless sensor networks. *China Commun.* **2012**, *9*, 103–111.
19. He, D.J.; Gao, Y.; Chan, S.; Chen, C.; Bu, J.J. An enhanced two-factor user authentication scheme in wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* **2010**, *10*, 361–371.
20. Das, A.K.; Sharma, P.; Chatterjee, S.; Sing, J.K. A dynamic password-based user authentication scheme for hierarchical wireless sensor networks. *J. Netw. Comput. Appl.* **2012**, *35*, 1646–1656. [CrossRef]
21. Turkanovic, M.; Hölbl, M. An improved dynamic password-based user authentication scheme for hierarchical wireless sensor networks. *Elektron. Elektrotech.* **2013**, *19*, 109–116. [CrossRef]
22. Turkanović, M.; Brumen, B.; Hölbl, M. A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion. *Ad Hoc Netw.* **2014**, *20*, 96–112. [CrossRef]
23. Wang, D.; Wang, P. Understanding security failures of two-factor authentication schemes for real-time applications in hierarchical wireless sensor networks. *Ad Hoc Netw.* **2014**, *20*, 1–15. [CrossRef]
24. Xue, K.P.; Ma, C.S.; Hong, P.L.; 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]
25. 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.* **2014**, *8*, 1070–1081. [CrossRef]
26. 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]

27. Gope, P.; Hwang, T. A Realistic Lightweight Anonymous Authentication Protocol for Securing Real-Time Application Data Access in Wireless Sensor Networks. *IEEE Trans. Ind. Electron.* **2016**, *63*, 7124–7132. [CrossRef]
28. Lu, Y.; Li, L.; Peng, H.; Yang, Y. An Energy Efficient Mutual Authentication and Key Agreement Scheme Preserving Anonymity for Wireless Sensor Networks. *Sensors* **2016**, *16*, 837. [CrossRef] [PubMed]
29. Jung, J.; Kim, J.; Choi, Y.; Won, D. An Anonymous User Authentication and Key Agreement Scheme Based on a Symmetric Cryptosystem in Wireless Sensor Networks. *Sensors* **2016**, *16*, 1299. [CrossRef] [PubMed]
30. Mir, O.; Munilla, J.; Kumari, S. Efficient anonymous authentication with key agreement protocol for wireless medical sensor networks. *Peer-to-Peer Netw. Appl.* **2017**, *10*, 79–91. [CrossRef]
31. Wang, D.; Wang, N.; Wang, P.; Qing, S. Preserving privacy for free: Efficient and provably secure two-factor authentication scheme with user anonymity. *Inf. Sci.* **2015**, *321*, 162–178. [CrossRef]
32. Wang, D.; He, D.; Wang, P.; Chu, C.H. Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment. *IEEE Trans. Dependable Secur. Comput.* **2015**, *12*, 428–442. [CrossRef]
33. Park, D.; Boyd, C.; Moon, S.J. Forward Secrecy and Its Application to Future Mobile Communications Security. In Proceedings of the Public Key Cryptography, Melbourne, Australia, 18–20 January 2000; pp. 433–445.
34. Ma, C.; Wang, D.; Zhao, S. Security Flaws in Two Improved Remote User Authentication Schemes Using Smart Cards. *Int. J. Commun. Syst.* **2012**, *27*, 2215–2227. [CrossRef]
35. Jiang, Q.; Zeadally, S.; Ma, J.; He, D. Lightweight Three-Factor Authentication and Key Agreement Protocol for Internet-Integrated Wireless Sensor Networks. *IEEE Access* **2017**, *5*, 3376–3392. [CrossRef]
36. Wang, D.; Wang, P. On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions. *Comput. Netw.* **2014**, *73*, 41–57. [CrossRef]
37. Fabrega, F.; Herzog, J.C.; Guttman, J.D. Strand spaces: Why is a security protocol correct? In Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, 6 May 1998; pp. 160–171.
38. Fabrega, F.; Herzog, J.C. Strand spaces: Proving Security Protocol Protocols Correct. *J. Comput. Secur.* **1999**, *7*, 191–230. [CrossRef]
39. Abadi, M.; Blanchet, B. Computer-assisted verification of a protocol for certified email. *Sci. Comput. Program.* **2005**, *58*, 3–27. [CrossRef]
40. Abadi, M.; Glew, N.; Horne, B.; Pinkas, B. Certified email with a light on-line trusted third party: Design and implementation. In Proceedings of the 11th International World Wide Web Conference, Honolulu, HI, USA, 7–11 May 2002; ACM: Honolulu, HI, USA, 2002; pp. 387–395.
41. Abadi, M.; Blanchet, B.; Cedric, F. Just Fast Keying in the pi calculus. *ACM Trans. Inf. Syst. Secur.* **2007**, *10*, 9. [CrossRef]
42. Blanchet, B.; Smyth, B.; Cheval, V.; Sylvestre, M. ProVerif 1.96: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial. Available online: http://prosecco.gforge.inria.fr/personal/bblanche/proverif/ (accessed on 21 September 2016).
43. Xiong, L. Lightweight-Anonymous-Authentication-WSNs. Available online: https://github.com/idle010/Lightweight-Anonymous-Authentication-WSNs (accessed on 26 October 2017).
44. Wang, D.; Wang, P. On the usability of two-factor authentication. In Proceedings of the 10th International Conference on Security and Privacy in Communication Systems, Beijing, China, 24–26 September 2014; pp. 141–150.
45. Wang, D.; Wang, P. Two Birds with One Stone: Two-Factor Authentication with Security beyond Conventional Bound. *IEEE Trans. Dependable Secur. Comput.* **2016**, 1–23. [CrossRef]
46. He, D.; Kumar, N.; Lee, J.H.; Sherratt, R. Enhanced three-factor security protocol for consumer USB mass storage devices. *IEEE Trans. Consum. Electron.* **2014**, *60*, 30–37. [CrossRef]
47. Lee, C.C.; Chen, C.T.; Wu, P.H.; Chen, T.Y. Three-factor control protocol based on elliptic curve cryptosystem for universal serial bus mass storage devices. *IET Comput. Digit. Tech.* **2013**, *7*, 48–55. [CrossRef]

© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).