Sensors **2015**, *15*, 14960-14980; doi:10.3390/s150714960

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Article

Efficient and Secure Temporal Credential-Based Authenticated Key Agreement Using Extended Chaotic Maps for Wireless Sensor Networks

Tian-Fu Lee

Department of Medical Informatics, Tzu Chi University, No. 701, Zhongyang Road, Sec. 3, Hualien 97004, Taiwan; E-Mail: jackytflee@mail.tcu.edu.tw; Tel.: +886-3-856-5301 (ext. 2403); Fax: +886-3-857-9409

Academic Editor: Leonhard M. Reindl

Received: 8 April 2015 / Accepted: 20 June 2015 / Published: 25 June 2015

Abstract: A secure temporal credential-based authenticated key agreement scheme for Wireless Sensor Networks (WSNs) enables a user, a sensor node and a gateway node to realize mutual authentication using temporal credentials. The user and the sensor node then negotiate a common secret key with the help of the gateway node, and establish a secure and authenticated channel using this common secret key. To increase efficiency, recent temporal credential-based authenticated key agreement schemes for WSNs have been designed to involve few computational operations, such as hash and exclusive-or operations. However, these schemes cannot protect the privacy of users and withstand possible attacks. This work develops a novel temporal credential-based authenticated key agreement scheme for WSNs using extended chaotic maps, in which operations are more efficient than modular exponential computations and scalar multiplications on an elliptic curve. The proposed scheme not only provides higher security and efficiency than related schemes, but also resolves their weaknesses.

Keywords: authentication; privacy protection; key agreement; temporal credential; wireless sensor networks; chaotic maps

1. Introduction

Wireless sensor networks (WSNs) comprise a large number of sensor nodes, and are utilized in many environments, such as dangerous areas in which humans must be medically monitored, military

environments in which reconnaissance and communication must be carried out, and others. Owing to the hardware limitations, sensor nodes in WSNs cannot support heavy computation loads, extensive communications or extensive storage. Thus, developing a lightweight and secure authenticated key agreement scheme is very important for WSNs. Temporal credential-based authenticated key agreements enable communicating entities to authenticate each other and to establish a secure and authenticated channel by confirming their temporal credentials. A temporal credential-based authenticated key agreement scheme for WSNs is composed of three classes of entity—users, sensor nodes and a gateway node (*GWN*)—and has registration, login, authentication and key agreement, and password change phases. In the registration phase, users and sensor nodes register their secret keys to the *GWN*. Then the *GWN* issues one temporal credential to each user and sensor node for authentication. In the login, authentication and key agreement phases, the user, the sensor node and *GWN* authenticate each other using these temporal credentials. Additionally, the user and the each sensor node negotiate a common secret key with the help of *GWN* to establish a secure and authentication channel in the WSN. Finally, the password change phase enables users to update their passwords for increased security [1–9].

Recently, Xue *et al.* [8] presented the concept of temporal credentials and developed a lightweight temporal credential-based authenticated key agreement scheme for WSNs. The scheme of Xue *et al.* has a lower computational burden, less extensive communication needs and requires less storage than previous approaches, and tries to provide more functionality and higher security [10–17]. Later, Li *et al.* [9] noted that the scheme of Xue *et al.* fails to withstand stolen-verifier attacks, password guessing attacks, insider attacks and lost smartcard attacks, and so proposed an advanced temporal credential-based scheme for WSNs as an alternative. However, in the scheme of Li *et al.*, an adversary can derive users' identities, temporal credentials, verification values in the *GWN*'s verifier table and expiration time from revealed messages allowing the adversary to perform successful impersonation attacks and stolen verifier attacks, easily discovering the hidden identity of the sender of the request message. Moreover, the adversary can derive all previous session keys of users and sensor nodes, and thus access all transmitted secrets. Accordingly, these temporal credential-based schemes for WSNs fail to resist possible attacks and to protect the privacy of users.

1.1. Our Contributions

This work addresses the weaknesses of the scheme of Li *et al.* and proposes an efficient and secure temporal credential-based authenticated key agreement scheme for WSNs that uses extended chaotic maps, and involves operations that are more efficient than modular exponential computations and scalar multiplications on an elliptic curve [18–20]. The proposed scheme protects a user's identity using a temporary secret key of the user and the gateway node, which security is based on the extended chaotic maps-based Diffie-Hellman problem [21–27], and reduces the number of parameters concerning each user's identity and password such that an adversary cannot impersonate any user or communicate with the gateway node or the sensor nodes, even if the adversary has stolen the verifier table and obtained the user's private information. Additionally the ephemeral parameters are randomly selected and independent among executions of the scheme. Thus, the adversary cannot derive any

previous session keys of the user and the sensor node. The proposed scheme avoids the weaknesses of previous schemes, has higher security and lower computational cost.

1.2. Enhanced Chebyshev Polynomial and Extended Chaotic Maps

Recent investigations have demonstrated that cryptosystems that use chaotic map operations are more efficient than those that use modular exponential computations and scalar multiplications on elliptic curves. Additionally, enhanced Chebyshev polynomials also exhibit the semi-group property and the commutative property, and they are subject to the discrete logarithm problem and the Diffie-Hellman problem [21–27], which are described as follows.

1.2.1. Enhanced Chebyshev Polynomial

The enhanced Chebyshev polynomial $T_n(x)$ is a polynomial in x of degree n, defined by the following recurrence relation:

$$\begin{cases} T_0(x) = 1; \\ T_1(x) = x; \text{ and} \\ T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \bmod p, \text{for } n \geq 2 \end{cases} \quad (1)$$

where and p is a large prime number. The enhanced Chebyshev polynomials satisfy the semi-group property and are commutative under composition. Then:

$$T_r(T_s(x)) \equiv T_{rs}(x) \equiv T_s(T_r(x)) \bmod p \quad (2)$$

holds.

1.2.2. Extended Chaotic Map-Based Discrete Logarithm Problem

Given x, y and p, it is computationally infeasible to find the integer r satisfying:

$$y = T_r(x) \bmod p \quad (3)$$

1.2.3. Extended Chaotic Map-Based Diffie-Hellman Problem

Given $T_u(x)$, $T_v(x)$, $T(.)$, x and p, where u, $v \geq 2$, $x \in (-\infty, +\infty)$ and p is a large prime number, it is computationally infeasible to calculate:

$$T_{u \cdot v}(x) \equiv T_u(T_v(x)) \equiv T_v(T_u(x)) \bmod p \quad (4)$$

1.3. Organization of the Paper

The rest of this paper is organized as follows: Section 2 reviews the temporal credential-based scheme of Li *et al.* for WNSs and elucidates its weaknesses. Section 3 presents the proposed efficient and secure temporal credential-based authenticated key agreement scheme for WSNs using extended chaotic maps. Sections 4 and 5 present the results of evaluations of the security and performance of the scheme, respectively. Finally, Section 6 draws conclusions.

2. The Temporal Credential-Based Scheme of Li *et al.* and Its Weaknesses

This section presents the notation used in this study, briefly reviews the advanced temporal credential-based scheme for wireless sensor networks proposed by Li *et al.* [9], and finally states its weaknesses.

Assume that U_i denotes the i-th user of WSNs; S_j denotes the j-th sensor node; and GWN denotes the Gateway node in which U_i and S_j are registered. Table 1 lists the notations which are used throughout this paper.

Table 1. Notation.

ID_i, PW_i	Identity and password pair of user U_i
SID_j	Pre-configured identity of the sensor node S_j
K_{GWN_U}, K_{GWN_S}	The long-term secret keys only known to GWN.
p	A large prime number
TCR_i, TCR_j	A temporal credential issued by GWN to U_i / S_j
E_i	The expiration time of U_i's temporal credential.
$t_1,t_2,\ldots,t_6$	The timestamp values.
Δt	The expected time interval for the transmission delay.
$h(.)$	A collision free one-way hash function [28]
$A \rightarrow B:M$	A sends message M to B through a common channel.
$\oplus$	The exclusive-or (XOR) operation
$M_1 \| M_2$	Message M_1 concatenates to message M_2.

2.1. Review of the Temporal Credential-Based Scheme of Li et al.

In 2013, Li *et al.* [9] proposed an advanced temporal credential-based scheme for WSNs, which consists of pre-registration, registration, login, authentication and key agreement phases, which are described as follows.

2.1.1. Pre-Registration Phase

Each user U_i has a pair of identity ID^{pre}_i and password PW^{pre}_i. GWN stores $h(ID^{pre}_i||PW^{pre}_i)$ and ID^{pre}_i in its storage. Similarly, each sensor node S_j is pre-configured with its identity SID_j and a random number r_j and the hash value $h(SID_j||r_j)$. Then r_j and SID_j are stored on the GWN's storage.

2.1.2. Registration Phase

(1) Registration phase for users

Step 1: $U_i \rightarrow GWN$: $\{ID^{pre}_i, t_1, VI_i, CI_i, DI_i\}$

U_i selects his/her ID_i, password PW_i, and a random number r_i, computes and sends $\{ID^{pre}_i, t_1, VI_i, CI_i, DI_i\}$ to GWN, where $VI_i = h(t_1||h(ID^{pre}_i||PW^{pre}_i))$, $CI_i = h(ID^{pre}_i||PW^{pre}_i) \oplus h(ID_i||PW_i||r_i)$, $DI_i = ID_i \oplus h(ID^{pre}_i||PW^{pre}_i)$ and t_1 is the current timestamp.

Step 2: $GWN \rightarrow U_i$: $\{h(Q_i)$, smartcard$\}$

GWN checks the validity of t_1, retrieves $h(ID^{pre}_i||PW^{pre}_i)$ by using ID^{pre}_i, computes

VI_i^* = $h(t_1||h(ID^{pre}_i||PW^{pre}_i))$ and checks VI_i^* =? VI_i. Then *GWN* computes $Q_i = CI_i \oplus h(ID^{pre}_i||PW^{pre}_i) = h(ID_i||PW_i||r_i)$, $DI_i = ID_i \oplus h(ID^{pre}_i||PW^{pre}_i)$, $P_i = h(ID_i||E_i)$, $TCR_i = h(K_{GMN\_U}||P_i||E_i)$ and $PTC_i = TCR_i \oplus Q_i$ and personalizes the smart card for U_i with the parameters: $\{h(.), h(Q_i), E_i, PTC_i\}$. *GWN* maintains a write protected file, where the status-bit indicates the status of the user, *i.e.*, when U_i is logged-in to *GWN*, the status-bit is 1, otherwise it is 0. Finally, *GWN* sends $h(Q_i)$ and smart card to U_i.

Step 3: U_i and authenticates *GWN* by checking $h(h(ID_i||PW_i||r_i))$ =? $h(Q_i)$ and enters r_i into his/her smart card. Then the smart card contains $\{h(.), h(Q_i), E_i, PTC_i, r_i\}$.

(2) Registration phase for sensor nodes

Step 1: $S_j \rightarrow GWN$: $\{SID_j, t_2, VI_j\}$
S_j computes $VI_j = h(t_2||h(SID_j||r_j))$ and sends $\{SID_j, t_2, VI_j\}$ to *GWN*, where t_2 is the current timestamp.

Step 2: $GWN \rightarrow S_j$: $\{t_3, Q_j, REG_j\}$
GWN checks the validity of t_2, retrieves $h(SID_j||r_j)$ by using SID_j and computes $VI_j^* = h(t_2||h(SID_j||r_j))$, checks VI_j^* =? VI_j, computes $TCR_j = h(K_{GMN\_S}||SID_j)$, $Q_j = h(t_3||h(SID_j||r_j))$ and $REG_j = h(h(SID_j||r_j)||t_3) \oplus TCR_j$, and sends $\{t_3, Q_j, REG_j\}$ to S_j, where t_3 is the current system timestamp.

Step 3: S_j checks the validity of t_3 and $h(t_3||h(SID_j||r_j))$ =? Q_j, computes its temporal credential $TCR_j = REG_j \oplus h(h(SID_j||r_j)||t_3)$ and stores it.

2.1.3. Login Phase

Step 1: U_i inserts his/her smart card into a card reader and enters ID_i and PW_i.

Step 2: The smartcard retrieves r_i, computes $Q_i' = h(ID_i||PW_i||r_i)$ and checks $h(Q_i')$ =? $h(Q_i)$. If successful, U_i passes the verification, allows to read the information stored in the smartcard, and computes $TCR_i = PTC_i \oplus Q_i'$.

2.1.4. Authentication and Key Agreement Phase

Step 1: $U_i \rightarrow GWN$: $\{DID_i, C_i, PKS_i, t_4, E_i, P_i\}$
U_i computes $DID_i = ID_i \oplus h(TCR_i||t_4)$, $C_i = h(h(ID_i||PW_i||r_i)||t_4) \oplus TCR_i$, $PKS_i = K_i \oplus h(TCR_i||t_4||"000")$, and sends $\{DID_i, C_i, PKS_i, t_4, E_i, P_i\}$ to *GWN*, where t_4 is the current timestamp.

Step 2: $GWN \rightarrow S_j$: $\{t_5, DID_i, DID_{GWN}, C_{GWN}, PKS_{GWN}\}$
GWN checks the validity of t_4, computes $TCR_i^* = h(K_{GMN\_U}||P_i||E_i)$ and $ID_i = DID_i \oplus h(TCR_i^*||t_4)$ and retrieves U_i's password-verifier of $Q_i = h(ID_i||PW_i||r_i)$ by using ID_i. Then, *GWN* further computes $C_i^* = h(Q_i ||t_4) \oplus TCR_i^*$, verifies C_i^* =? C_i, sets the status-bit as "1" and records t_4 in the 4th field of the identity table. *GWN* computes $K_i = PKS_i \oplus h(TCR_i^*||t_4||"000")$ and chooses a nearby suitable sensor

node S_j as the accessed sensor node. *GWN* further computes S_j's temporal credential $TCR_j = h(K_{GWN\_S}||SID_j)$, $DID_{GWN} = ID_i \oplus h(DID_i||TCR_j||t_5)$, $C_{GWN} = h(ID_i||TCR_j||t_5)$ and $PKS_{GWN} = K_i \oplus h(TCR_i||t_5)$ and sends $\{t_5, DID_i, DID_{GWN}, C_{GWN}, PKS_{GWN}\}$ to S_j, where t_5 is the current timestamp of *GWN*.

Step 3: $S_j \rightarrow GWN, U_i$: $\{SID_j, t_6, C_j, PKS_j\}$

S_j checks the validity of t_5, computes $ID_i = DID_{GWN} \oplus h(DID_i||TCR_j||t_5)$ and $C_{GWN}^* = h(ID_i||TCR_j||t_5)$, and checks $C_{GWN}^* =? C_{GWN}$. If unsuccessful, S_j terminates this session; otherwise, S_j convinces that the received message is from a legitimate *GWN*. Moreover, S_j computes $K_i = PKS_{GWN} \oplus h(TCR_i||t_5)$, $C_j = h(K_j||ID_i||SID_j||t_6)$ and $PKS_j = K_j \oplus h(K_i||t_6)$ and sends $\{SID_j, t_6, C_j, PKS_j\}$ to *GWN* and U_i, where t_6 is the current timestamp of S_j.

Step 4: U_i and *GWN* separately computes $K_j = PKS_j \oplus h(K_i||t_6)$ and $C_j^* = h(K_j||ID_i||SID_j||t_6)$. *GWN* authenticates S_j by checking $C_j^* =? C_j$. U_i authenticates S_j and *GWN* by checking $C_j^* =? C_j$. Finally, U_i and S_j computes a common session key $K_{ij} = h(K_i||K_j)$ for later securing communications.

2.2. Weaknesses of Temporal Credential-Based Scheme of Li et al.

This subsection elucidates the weaknesses of the temporal credential-based scheme of Li *et al.*, which include vulnerability to impersonation and stolen verifier attacks, and failure to protect the privacy of users.

2.2.1. Vulnerability to Impersonation Attacks

In the registration phase of the scheme of Li *et al.*, since $(ID^{pre}_i, t_1, VI_i, CI_i, DI_i)$ and $(h(.), h(Q_i), E_i, PTC_i)$ are public, where $VI_i = h(t_1||h(ID^{pre}_i||PW^{pre}_i))$, $CI_i = h(ID^{pre}_i||PW^{pre}_i) \oplus h(ID_i||PW_i||r_i)$, $DI_i = ID_i \oplus h(ID^{pre}_i||PW^{pre}_i)$ and t_1 is the current timestamp, an adversary, $\mathcal{A}$, can obtain a correct PW^{pre}_i by guessing a password PW^{pre*}_i and checking $VI_i = ? h(t_1||h(ID^{pre}_i||PW^{pre*}_i))$ repeatedly. Next, the adversary can derive ID_i, Q_i ($=h(ID_i||PW_i||r_i)$) and TCR_i by computing $DI_i \oplus h(ID^{pre}_i||PW^{pre}_i)$, $h(ID^{pre}_i||PW^{pre}_i) \oplus CI_i$ and $PTC_i \oplus Q_i|$, respectively. $\mathcal{A}$ can subsequently impersonate U_i and compromise U_i's privacy based on knowledge of (ID_i, Q_i, TCR_i, E_i). By the following steps, $\mathcal{A}$ can successfully impersonate U_i, be authenticated, and communicate with *GWN* and S_j:

Step 1: First, the adversary $\mathcal{A}$ retrieves P_i using E_i. In the authentication and key agreement phase, $\mathcal{A}$ can compute $DID_i = ID_i \oplus h(TCR_i||t_4)$, $C_i = h(h(Q_i||t_4) \oplus TCR_i)$, $PKS_i = K_i \oplus h(TCR_i||t_4||"000")$, where t_4 is the current timestamp. Then, $\mathcal{A}$ successfully impersonates U_i and sends $\{DID_i, C_i, PKS_i, t_4, E_i, P_i\}$ to *GWN*.

Step 2: *GWN* checks t_4, computes $TCR_i^* = h(K_{GWN\_U}||P_i||E_i)$ and $ID_i = DID_i \oplus h(TCR_i^*||t_4)$, $C_i^* = h(h(Q_i||t_4) \oplus TCR_i^*)$ and verifies $C_i^* =? C_i^*$. Then, *GWN* computes $K_i = PKS_i \oplus h(TCR_i||t_4||"000")$, $TCR_j = h(K_{GWN\_S}||SID_j)$, $DID_{GWN} = ID_i \oplus h(DID_i||TCR_j||t_5)$,

$C_{GWN} = h(ID_i||TCR_j||t_5)$ and $PKS_{GWN} = K_i \oplus h(TCR_i||t_5)$ and sends $\{t_5, DID_i, DID_{GWN}, C_{GWN}, PKS_{GWN}\}$ to S_j, where t_5 is the current timestamp of *GWN*.

Step 3: S_j checks t_5, computes $ID_i = DID_{GWN} \oplus h(DID_i||TCR_j||t_5)$, $C_{GWN}^* = h(ID_i||TCR_j||t_5)$, $K_i = PKS_{GWN} \oplus h(TCR_i||t_5)$and $C_j = h(K_j||ID_i||SID_j||t_6)$; verifies $C_{GWN}^* =? C_{GWN}$, and responds by sending $\{SID_j, t_6, C_j, PKS_j\}$ to *GWN* and $\mathcal{A}$, where $PKS_j = K_j \oplus h(K_i||t_6)$. Finally, $\mathcal{A}$ computes $K_j = PKS_j \oplus h(K_i||t_6)$ and shares the common session key $K_{ij} = h(K_i||K_j)$ with S_j.

However, if the password PW^{pre}_i is sufficiently long, the credential based key agreement scheme of Li, *et al*. can resist the impersonation attacks.

2.2.2. Failure to Protect the Privacy of Users

In the scheme of Li *et al*., upon receiving the request message $\{DID_i, C_i, PKS_i, t_4, E_i, P_i\}$ that is sent by U_i, whose identity is ID_i, the adversary $\mathcal{A}$ easily determines that the request message belongs to U_i because $\mathcal{A}$ has the knowledge of (ID_i, Q_i, TCR_i, E_i). Thus, the scheme of Li *et al.* fails to support user anonymity, data unlinkability, or untrackability [29]. Accordingly, the scheme of Li *et al.* cannot protect the privacy of users.

2.2.3. Vulnerability to Stolen Verifier Attacks

Assume that an adversary $\mathcal{A}$ steals the verifier table and obtains (ID_i, Q_i, E_i). The adversary $\mathcal{A}$ can derive TCR_i using $PTC_i \oplus Q_i$, since ($h(.)$, $h(Q_i)$, E_i, PTC_i) is public in the registration phase:

Step 1: $\mathcal{A} \rightarrow GWN$: $\{DID_i^{**}, C_i\ t_4^{**}, PKS_i, t_4^{**}, E_i, P_i\}$

$\mathcal{A}$ randomly selects K_i^{**}, computes $DID_i^{**} = ID_i \oplus h(TCR_i||t_4^{**})$, $C_i^{**} = h(Q_i||t_4^{**}) \oplus TCR_i$ and $PKS_i^{**} = K_i^{**} \oplus h(TCR_i||t_4^{**}||"000")$, where t_4^{**} is the current timestamp, and sends $\{DID_i^{**}, C_i\ t_4^{**}, PKS_i, t_4^{**}, E_i, P_i\}$ to *GWN*.

Step 2: $GWN \rightarrow S_j$: $\{t_5, DID_i^{**}, DID_{GWN}, C_{GWN}, PKS_{GWN}\}$

GWN validates t_4^{**}, computes $TCR_i^* = h(K_{GMN\_U}||P_i||E_i)$ and $ID_i = DID_i^{**} \oplus h(TCR_i^*||t_4^{**})$, and retrieves $Q_i = h(ID_i||PW_i||r_i)$. Then, *GWN* verifies $h(Q_i\ ||t_4^{**}) \oplus TCR_i^* = C_i^{**}$, computes $K_i = PKS_i \oplus h(TCR_i^*||t_4^{**}||"000")$, $TCR_j = h(K_{GWN\_S}||SID_j)$, $DID_{GWN} = ID_i \oplus h(DID_i^{**}||TCR_j||t_5)$, $C_{GWN} = h(ID_i||TCR_j||t_5)$ and $PKS_{GWN} = K_i \oplus h(TCR_i||t_5)$, and sends $\{t_5, DID_i^{**}, DID_{GWN}, C_{GWN}, PKS_{GWN}\}$ to S_j, where t_5 is the current timestamp of *GWN*.

Step 3: $S_j \rightarrow GWN, U_i$: $\{SID_j, t_6, C_j, PKS_j\}$

S_j validates t_5. If successful, S_j computes $ID_i = DID_{GWN} \oplus h(DID_i^{**}||TCR_j||t_5)$ and $C_{GWN}^* = h(ID_i||TCR_j||t_5)$ and checks $C_{GWN}^* =? C_{GWN}$, computes $K_i^{**} = PKS_{GWN} \oplus h(TCR_i||t_5)$, $C_j = h(K_j||ID_i||SID_j||t_6)$ and $PKS_j = K_j \oplus h(K_i^{**}||t_6)$ and sends out $\{SID_j, t_6, C_j, PKS_j\}$.

Step 4: Upon receiving $\{SID_j, t_6, C_j, PKS_j\}$, $\mathcal{A}$ computes $K_j = PKS_j \oplus h(K_i^{**}||t_6)$ and a common session key $K_{ij} = h(K_i||K_j)$ that is shared with S_j.

Hence, the adversary $\mathcal{A}$ can impersonate U_i, be authenticated, and communicate with *GWN* and S_j. Additionally, $\mathcal{A}$ has TCR_i and messages (PKS_i, t_4) and (PKS_j, t_6), which were previously sent out by user U_i. A can therefore derive previous secrets K_i and K_j by computing $PKS_i \oplus h(TCR_i||t_4||"000")$ and $PKS_i \oplus h(K_i||t_6)$, respectively. $\mathcal{A}$ can calculate all session keys that have been used by U_i and S_j, and thereby derive all transmitted secrets. Therefore, the authenticated key agreement scheme of Li *et al.* fails to resist stolen verifier attacks.

3. Proposed Temporal Credential-Based Scheme Using Chaotic Maps for WSNs

This section describes the use of chaotic maps in a new temporal credential-based authenticated key agreement scheme for WSNs. The novel scheme does not reveal the user's private parameters in the registration phase, and it protects the user's identity with a temporary secret key of the user and the gateway node. The security of this temporary secret key is based on the extended chaotic map-based Diffie-Hellman problem. The proposed approach also reduces the redundant parameters associated with the user's identity and password, which are stored in the *GWN*'s verifier table, preventing an adversary from impersonating a user and communicating with the gateway node and sensor nodes, even if the adversary has stolen the verifier table and obtained the user's private information. The session key security is based on the extended chaotic map-based Diffie-Hellman problem, so the adversary cannot derive any previous session key of the user and the sensor node. In the proposed scheme, the user does not know which node it can access and communicate with, thus *GWN* requires choosing a nearby suitable sensor node as the accessed sensor node. The proposed scheme involves parameter generation, pre-registration, registration, login and authentication and password change phases, which are described below.

3.1. Parameter Generation Phase

Step 1: The gateway node *GWN* randomly selects K_{GWN} as its master secret key.

Step 2: *GWN* computes $PK_G = T_{K_{GWN}}(x) \bmod p$, where x is a random number, p is a large prime number and (PK_G, $T(.)$, x, p) are public parameters.

3.2. Pre-Registration Phase

Each user U_i has a pre-configured identity ID^{pre}_i, which is stored in the *GWN*'s storage. Similarly, each sensor node S_j is pre-configured with its identity SID_j and a random number r_j and the hash value $h(SID_j \| r_j)$. Then $h(SID_j \| r_j)$ and SID_j are stored on the *GWN*'s storage. The pre-configured data is transferred by using physical delivery.

3.3. Registration Phase

3.3.1. Registration Phase for Users

Step 1: $U_i \rightarrow GWN$: $\{X_0, X_1, REG_i, t_1\}$

U_i chooses his/her identity ID_i, password PW_i, random numbers r and r_i, and computes $K_{UG} = T_r(PK_G)$ mod p, $X_0 = T_r(x)$ mod p, $REG_i = K_{UG} \oplus (ID^{pre}{}_i \| ID_i \| h(ID_i \| PW_i \| r_i)$, and $X_1 = h(K_{UG} \| h(ID_i \| PW_i \| r_i) \| t_1)$, where t_1 is the current timestamp. Then U_i sends $\{X_0, X_1, REG_i, t_1\}$ to GWN.

Step 2: $GWN \rightarrow U_i$: $\{Y_0, Y_1\}$

Upon receiving the register message form U_i, GWN checks the validity of t_1 and computes $K_{UG} = T_{K_{GWN}}(X_0)$ mod p and $ID^{pre}{}_i \| ID_i \| h(ID_i \| PW_i \| r_i) = REG_i \oplus K_{UG}$, and extracts $(ID^{pre}{}_i, ID_i, h(ID_i \| PW_i \| r_i))$. If GWN successfully checks $h(K_{UG} \| h(ID_i \| PW_i \| r_i) \| t_1) =? X_1$ and verifies that $ID^{pre}{}_i$ is in GWN's storage and has not been registered, then generates an expiration time E_i, and computes U_i's temporal credential $TCR_i = h(K_{GMN} || ID_i || E_i)$, $D_1 = TCR_i \oplus h(ID_i \| PW_i \| r_i)$, $Y_0 = D_1 \oplus h(K_{UG} \| t_1)$ and $Y_1 = h(D_1 \| K_{UG} \| t_1)$. Then, GWN sends $\{Y_0, Y_1\}$ to U_i. GWN also stores $(h(ID_i), E_i)$ in its storage and maintains a status-bit b and a last login field to indicate the status of the user. If U_i logins GWN, $b = 1$, otherwise $b = 0$.

Step 3: After receiving the response message form GWN, U_i computes $D_1 = Y_0 \oplus h(K_{UG} \| t_1)$, checks $h(D_1 \| K_{UG} \| t_1) =? Y_1$. If successful, U_i inserts $(D_1, PK_G, T(.), x, p, h(.), r_i)$ into a smartcard and finishes the registration.

3.3.2. Registration Phase for Sensor Nodes

Step 1: $S_j \rightarrow GWN$: $\{SID_j, Z_0, t_2\}$

S_j computes $REG_j = h(SID_j \| r_j)$, $Z_0 = h(REG_j \| t_2)$, and sends $\{SID_j, Z_0, t_2\}$ to GWN, where t_2 is the current timestamp.

Step 2: $GWN \rightarrow S_j$: $\{SID_j, Y_2, Y_3\}$

Upon receiving $\{SID_j, Z_0, t_2\}$, GWN successfully checks the validity of t_2 and $h(REG_j || t_2) =? Z_0$ and verifies that SID_j has not been registered, then computes S_j's temporal credential $TCR_j = h(K_{GWN} \| REG_j)$, $Q_j = TCR_j \oplus REG_j$, $Y_2 = TCR_j \oplus h(t_2 \| REG_j)$, $Y_3 = h(TCR_j \| REG_j \| t_2)$ stores (SID_j, Q_j) in its storage, and sends $\{SID_j, Y_2, Y_3\}$ to S_j.

Step 3: S_j computes its temporal credential $TCR_j = Y_2 \oplus h(t_2 \| REG_j)$, checks $h(TCR_j \| REG_j \| t_2) =? Y_3$, and stores $(SID_j, TCR_j, REG_j, T(.), x, p, h(.))$ in its storage.

3.4. Login and Authentication Phase

In this phase, as shown in Figure 1, U_i and GWN authenticate each other by performing the following steps:

Step 1: $U_i \rightarrow GWN$: $M_1 = \{DID_i, X_2, X_3, t_3\}$

U_i inserts his smart card, inputs ID_i, and PW_i, computes $TCR_i = D_1 \oplus h(ID_i \| PW_i \| r_i)$, generates a random number u, calculates $K_1 = T_u(PK_G) \bmod p$, $DID_i = ID_i \oplus K_1$ and $X_2 = T_u(x) \bmod p$, $X_3 = h(ID_i \| K_1 \| TCR_i \| t_3)$, where t_3 is the current timestamp, and sends $M_1 = \{DID_i, X_2, X_3, t_3\}$ to GWN.

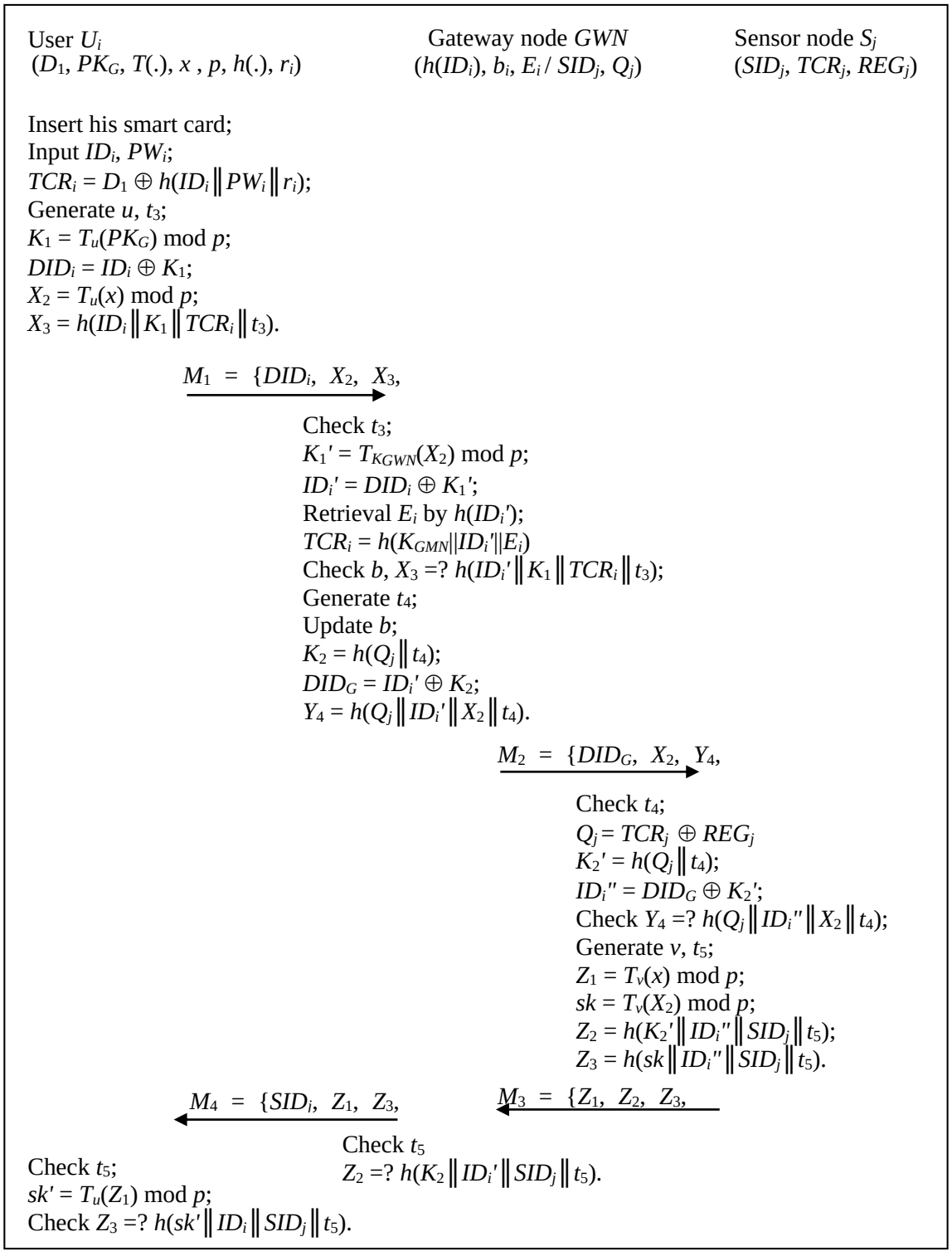

Figure 1. The login and authentication phase of the proposed scheme for WSNs.

Step 2: $GWN \rightarrow S_j$: $M_2 = \{DID_G, X_2, Y_4, t_4\}$
Upon receiving M_1, GWN checks the validity of t_3. If unsuccessful, GWN rejects this service request; Otherwise GWN computes $K_1' = T_{K_{GWN}}(X_2)$ mod p, $ID_i' = DID_i \oplus K_1'$, retrieval E_i by $h(ID_i')$, computes $TCR_i = h(K_{GMN}||ID_i'||E_i)$, and checks the status-bit, $X_3 =? h(ID_i' \| K_1 \| TCR_i \| t_3)$. If unsuccessful, GWN rejects this service request; Otherwise GWN updates the status-bit, and chooses an accessed sensor node sensor node S_j which is nearby and suitable, computes $K_2 = h(Q_j \| t_4)$, $DID_G = ID_i' \oplus K_2$, $Y_4 = h(Q_j \| ID_i' \| X_2 \| t_4)$, where t_4 is the current timestamp, and sends $M_2 = \{DID_G, X_2, Y_4, t_4\}$ to S_j.

Step 3: $S_j \rightarrow GWN$: $M_3 = \{Z_1, Z_2, Z_3, t_5\}$
Upon receiving M_2, S_j checks the validity of t_4. If unsuccessful, S_j aborts this service request; Otherwise S_j computes $Q_j = TCR_j \oplus REG_j$, $K_2' = h(Q_j \| t_4)$, $ID_i'' = DID_G \oplus K_2'$, and checks $Y_4 =? h(Q_j \| ID_i'' \| X_2 \| t_4)$. If unsuccessful, S_j still aborts this service request; Otherwise, S_j generates v, calculates $Z_1 = T_v(x)$ mod p, $sk = T_v(X_2)$ mod p, $Z_2 = h(K_2' \| ID_i'' \| SID_j \| t_5)$, $Z_3 = h(sk \| ID_i'' \| SID_j \| t_5)$, where t_5 is the current timestamp, and sends $M_3 = \{Z_1, Z_2, Z_3, t_5\}$ to GWN.

Step 4: $GWN \rightarrow U_i$: $M_4 = \{SID_j, Z_1, Z_3, t_5\}$
Upon receiving M_3, GWN checks the validity of t_5. If unsuccessful, GWN rejects this request; Otherwise, GWN authenticates S_j by checking $Z_2 =? h(K_2 \| ID_i' \| SID_j \| t_5)$, and sends $M_4 = \{SID_j, Z_1, Z_3, t_5\}$ to U_i.

Step 5: Upon receiving M_4, U_i checks the validity of t_5. If unsuccessful, U_i aborts this request; Otherwise, U_i computes $sk' = T_u(Z_1)$ mod p and authenticates GWN and S_j by checking $Z_3 =? h(sk' \| ID_i \| SID_j \| t_5)$. Finally, U_i and S_j obtain a common session key $sk = T_{uv}(x)$ mod p for later securing communications.

3.5. Password Change Phase

A user U_i changes his/her password by performing the following steps:

Step 1: U_i inserts his smart card and inputs his/her identity ID_i, old password PW_i, and a new password PW_i'.

Step 2: The smart card computes $Q_i = h(ID_i \| PW_i \| r_i)$ and $Q_i' = h(ID_i \| PW_i' \| r_i)$ and $D_1' = D_1 \oplus Q_i \oplus Q_i'$. Then the smart card replaces D_1 with D_1'.

4. Security Analyses

This section analyzes the security of the proposed authenticated key agreement scheme, which provides mutual authentication, session key security and privacy protection for users, and resists potential attacks, including privileged insider attacks, password guessing attacks, impersonation attacks, stolen verifier attacks and many-logged-in-users attacks. The details are described below.

4.1. Communication Model

4.1.1. Communicating Participants:

The proposed scheme involves a user U_i, a sensor node S_j, and a gateway node *GWN*. U_i and S_j authenticate each other and establish a common session key *sk* with the help of the *GWN*. A participant may be involved in several instances, called oracles, of distinct concurrent executions of the proposed scheme **P**. The instance *m* of participant *V* is denoted as ΠV^m.

4.1.2. Oracle Queries:

Oracle queries model the capabilities of adversary $\mathcal{A}$, and are described below:

(1) ***Send***(ΠV^m, *M*): This query models the capacity of an adversary $\mathcal{A}$ to control all communications in **P**. $\mathcal{A}$ sends a message *M* to oracle ΠV^m; then ΠV^m sends back a response message using **P**. $\mathcal{A}$ can initiate the execution of **P** by sending a query (ΠV^m, "*start*") to a user oracle ΠV^m.

(2) ***Corrupt***(*V*): This query models the perfect forward secrecy of **P**, meaning that a compromised long-lived key fails to endanger previous session keys. The adversary $\mathcal{A}$ sends a corrupt query to a participant *V*, and returns *V*'s long-life key.

(3) ***Hash***(*M*): This query models adversary $\mathcal{A}$'s reception of hash results by sending queries to a random oracle Ω. Upon receiving a query, Ωchecks whether a record (*M*, *r*) has been queried and recorded in the **H**-*table*. If (*M*, *r*) in the **H**-*table*, then Ω replies *r* to A; otherwise it returns a nonce *r'*, and keeps (*M*, *r'*) in the **H**-*table*.

(4) ***Reveal***(ΠV^m): This query models the known key security of **P**: a compromised session key fails to reveal other session keys, and is only available if oracle ΠV^m has accepted.

(5) ***Test***(ΠV^m): This query models the session key security to determine the indistinguishability of the real session key from a random string. During the execution of scheme **P**, adversary $\mathcal{A}$ sends queries to the oracle, including a single ***Test*** query at any time. Then, ΠV^m flips an unbiased coin *c*. If *c* equals 1, then ΠV^m returns the real session key *sk*; otherwise, it returns a random string to $\mathcal{A}$.

4.2. Security Definitions

4.2.1. Partnering: Two user oracles ΠU_i^m and ΠS_j^n are partnered if:

(1) ΠU_i^m and ΠS_j^n directly exchange message flows and
(2) only ΠU_i^m and ΠS_j^n have the same session key *sk*.

4.2.2. Freshness: An Oracle ΠU_i^m is **Fresh** in **P** if:

(1) ΠU_i^m or ΠS_j^n has accepted a session key *sk* and
(2) ΠU_i^m and ΠS_j^n have not been sent a **Reveal** query.

4.2.3. Session Key Security (AKE Security):

This definition allows an adversary to generate many ***Test*** queries. If a ***Test*** query is generated concerning a client instance that has not *accepted*, then the invalid symbol ⊥ is returned. If a ***Test*** query is generated concerning an instance of an honest participant whose intended partner is dishonest or an instance of a dishonest participant, then replies with the real session key. Otherwise, the reply to the ***Test*** query provides either the real session key or a random string, as determine by flipping an unbiased coin, *c*. The adversary seeks to guess correctly the value of the hidden bit *c* that is used by the Test oracle. The *ake-advantage* of the event that an adversary violates the indistinguishability of scheme **P** is denoted as $Adv_P^{ake}(\mathcal{A})$. The scheme **P** is AKE-secure if $Adv_P^{ake}(\mathcal{A})$ is negligible [30–32].

4.2.4. Mutual Authentication (MA Security)

In the execution of **P**, the adversary $\mathcal{A}$ violates mutual authentication if A can fake the authenticator. The probability of this event is denoted by $Adv_P^{ma}(\mathcal{A})$. The scheme **P** is MA-secure if $Adv_P^{ma}(\mathcal{A})$ is negligible [33].

4.3. Providing Session Key Security (AKE Security)

The following lemma describes the Difference Lemma, which is made used within our sequence of games [34].

Lemma 1 (Difference Lemma). *Let A, B and F be events defined in some probability distribution, and suppose that* $A \wedge \neg F \Leftrightarrow B \wedge \neg F$. *Then*

$$|Pr[A] - Pr[B]| \leq Pr[F]$$

The following theorem shows that the proposed scheme involving U_i *and* S_j *has AKE security if the used hash function is secure and the extended chaotic map-based Diffie-Hellman assumption holds*.

Theorem 1. *Let* Adv^{ecmdh} *be the advantage that an ECMDH attacker solves the extended chaotic map-based Diffie-Hellman problem within time t. Then, the probability that an adversary breaks the AKE security of the proposed scheme:*

$$Adv_P^{ake}(t', q_{exe}, q_{test}, q_{se}, q_{ake}) \leq 2 \cdot Adv^{ecmdh}(t, q_{test}, q_{se}, q_{ake})$$

within time t' *and* $t' \leq t + 4(q_{exe}+q_{ake})\tau$, *where* q_{exe} *denotes the number of queries to the Execute oracle;* q_{test} *denotes the number of queries to the Test oracle;* q_{se} *denotes the numbers of the Send queries;* q_{ake} *denotes the number of queries to the final AKE scheme; and* τ *is the time to perform an extended chaotic map operation.*

Proof of Theorem 1. Each game G_i defines the probability of the event E_i that the adversary wins this game. The first game G_0 is the real attack against the proposed scheme and the final game G_2 concludes that the adversary has a negligible advantage to break the AKE security of the proposed scheme:

Game G_0: This game corresponds to the real attack. By definition, we have

$$Adv_P^{ake}(\mathcal{A}) = |2\Pr[E_0] - 1| \quad (5)$$

Game G_1: This game simulates all oracles as in previous game except for modifying the simulation of ***Send*** queries refereeing the flows containing $T_u(x)$ mod p and $T_v(x)$ mod p of the proposed scheme, and the simulation of the **Test**(Πv^m) oracle to avoid relying on the knowledge of u, v and w used to compute the answer to these queries. Assume that (X, Y, Z) = ($T_u(x)$ mod p, $T_v(x)$ mod p, $T_{u \cdot v}(x)$ mod p) is a random extended chaotic map-based Diffie-Hellman triple. A simulator Σ simulates the oracles for all sessions by using this triple (X, Y, Z) and the classical random self-reducibility of the extended chaotic map-based Diffie-Hellman problem. Next, Σ sets up all parameters and secret keys of the scheme, and picks a random number $m \in [1, q_{se}]$ and answers the oracle queries according to the proposed scheme. Σ thus can correctly return the ***Test*** queries. Additionally, the random variables in G_0 is replaced by another random variables in G_1. Then we have that G_0 and G_1 is equivalent, and thus:

$$\Pr[E_0] = \Pr[E_1] \quad (6)$$

Game G_2: This game simulates all oracles as in previous game except that all rules are computed using a triple (X, Y, Z) from a random distribution ($T_u(x)$ mod p, $T_v(x)$ mod p, $T_w(x)$ mod p), instead of an extended chaotic map-based Diffie-Hellman triple. Let a challenger $\mathcal{A}_{ecdh}$ try to violate the indistinguishability of the extended chaotic map-based Diffie-Hellman problem; and an adversary $\mathcal{A}_{ake}$ be constructed to break the session key security. $\mathcal{A}_{ecdh}$ returns the real session key sk (if $c = 1$) or a random string (otherwise) to $\mathcal{A}_{ake}$ by flipping an unbiased coin $c \in \{0,1\}$. Then $\mathcal{A}_{ake}$ wins the game if its output bit c' equals c. $\mathcal{A}_{ecmdh}$ is asked ***Send***, ***Corrupt*** or ***Test*** queries, and returns the responses by using a previous experiment except for (X, Y, Z) that it had received as input. If $\mathcal{A}_{ake}$ outputs c, then $\mathcal{A}_{ecmdh}$ outputs 1; otherwise, $\mathcal{A}_{ecmdh}$ outputs 0. If (X, Y, Z) is a real extended chaotic map-based Diffie-Hellman triple, then $\mathcal{A}_{ecmdh}$ runs $\mathcal{A}_{ake}$ in G_1 and thus the probability of the event that $\mathcal{A}_{ecmdh}$ outputs 1 equals the probability of E_1. If (X, Y, Z) is a random triple, $\mathcal{A}_{ecmdh}$ runs $\mathcal{A}_{ake}$ in G_2 and thus the probability of the event that $\mathcal{A}_{ecdh}$ outputs 1 equals the probability of E_2. Therefore, we have:

$$|\Pr[E_1] - \Pr[E_2]| \leq Adv^{ecmdh}(\mathcal{A}_{ecmdh}) \quad (7)$$

Since the coin bit c and all sessions keys are random and independent, we have

$$\Pr[E_2] = 1/2 \quad (8)$$

By combining Equations (5)–(8) and using Lemma 1, we have:

$$Adv_P^{ake}(\mathcal{A}_{ake}) \leq 2 \cdot Adv^{ecmdh}(\mathcal{A}_{ecmdh})$$

Then the proof is concluded.

4.4. Providing Mutual Authentication

The following theorem shows that the proposed scheme has MA security if the used hash function is secure and the proposed scheme has AKE security:

Theorem 2. *Let* Adv_P^{ake} *denote the advantage that an adversary breaks the AKE security of the proposed scheme, and* Adv_P^{ma} *denote the advantage that an adversary violates the mutual authentication of the proposed scheme. Then:*

$$Adv_P^{ma}(t'', q_{se}, q_h) \le 2 \cdot Adv_P^{ake}(t', q_{se}, q_h) + q_h^2 / 2^{l-1}$$

within time t'' and $t'' \le t' + (q_{se} + q_h) \cdot t_{relay} + 2 \cdot \tau$, *where* q_h *denotes the numbers of the Hash queries;* t_{relay} *denotes the time to relay a query; l denotes the security parameter and the parameters* q_{se}, *t' and* τ *are defined as in Theorems 1.*

Proof of Theorem 2. The start game G^{ma}_0 is the real attack against the proposed scheme and the final game G^{ma}_2 concludes that the adversary has a negligible advantage to break MA security of the proposed scheme. The challenger $\mathcal{A}_1$ attempts to break AKE security of the proposed scheme and the adversary $\mathcal{A}_{ma}$ is constructed to break MA security of the proposed scheme. The adversary $\mathcal{A}_{ma}$ wins this game if he successfully fakes the authenticator:

Game G^{ma}_0: This game corresponds to the real attack. By definition, we have:

$$Adv_P^{ma}(\mathcal{A}_{ma}) = |2\Pr[E_0] - 1| \quad (9)$$

Game G^{ma}_1: This game simulates all oracles as in previous game except for using a table list **H** to simulate ***Hash*** queries involving U_i and *GWN*, and involving *GWN* and S_j. Then, games G^{ma}_0 and G^{ma}_1 are **undistinguishable** except collisions of **H**-*table* in G^{ma}_1. By using the birthday paradox and Lemma 1, we have:

$$|\Pr[E_0] - \Pr[E_1]| \le q_h^2 / 2^l \quad (10)$$

where $\mathcal{A}_{ma}$ makes q_h ***Hash*** queries involving U_i and *GWN*, and involving *GWN* and S_j.

Game G^{ma}_2: This game simulates all oracles as in previous game except for replacing the session key *sk* with a random number. Then, $\mathcal{A}_{ma}$ is used for building an adversary $\mathcal{A}_1$ against the AKE security of the proposed scheme. Next, $\mathcal{A}_1$ arranges the parameters, simulates the proposed scheme and replies the oracle queries made by $\mathcal{A}_{ma}$ by using following scenarios.

—When receiving ***Send*** or ***Hash*** queries involving U_i and *GWN*, and involving *GWN* and S_j, $\mathcal{A}_1$ replies the results by executing the proposed scheme.

—When receiving ***Hash*** queries involving U_i and S_j, $\mathcal{A}_1$ replies corresponding authenticators to $\mathcal{A}_{ma}$ by making the same queries to the oracle Hash involving U_i and S_j.

—When receiving ***Test*** queries, $\mathcal{A}_1$ replies these queries by using the coin bit *c* that it has previously selected and the computed session keys.

Therefore, the probability of the event that $\mathcal{A}_1$ outputs 1 when the authenticator is obtained by the real session key equals the probability of the event that $\mathcal{A}_{ma}$ correctly guesses the hidden bit *c* in game G^{ma}_1. Similarly, the probability that $\mathcal{A}_1$ outputs 1 when the authenticator obtained by a random string equals the probability that $\mathcal{A}_{ma}$ correctly guesses the hidden bit *c* in game G^{ma}_2. Thus, by Lemma 1, we have:

$$|\Pr[E_1] - \Pr[E_2]| \leq Adv_P^{ake}(\mathcal{A}_1) \quad (11)$$

Since no information on the authenticator is leaked to the adversary, we have

$$\Pr[E_2] = 1/2 \quad (12)$$

Combining Equations (9)–(12) and using Lemma 1, we have

$$Adv_P^{ma}(\mathcal{A}_{ma}) \leq 2 \cdot Adv_P^{ake}(\mathcal{A}_1) + q_h^2 / 2^{l-1}$$

Then the proof is concluded.

4.5. Protecting Privacy of Users

Theorem 3. *The proposed scheme protects the privacy of users.*

Proof of Theorem 3. The proposed scheme protects user U_i's identity ID_i using the temporary secret key K_1 of the user and the gateway node, and enables any two request messages $M_1 = \{DID_i, X_2, X_3, t_3\}$ and $M_1' = \{DID_i', X_2', X_3', t_3'\}$ from user U_i to be independent and difficult to distinguish from each other, where $K_1 = T_u(PK_G)$ mod p, $DID_i = ID_i \oplus K_1$, $X_2 = T_u(x)$ mod p, $X_3 = h(ID_i \| K_1 \| TCR_i \| t_3)$, u is a random number and t_3 is a timestamp; and $K_1' = T_{u'}(PK_G)$ mod p, $DID_i' = ID_i \oplus K_1'$, $X_2' = T_{u'}(x)$ mod p, $X_3' = h(ID_i \| K_1' \| TCR_i \| t_3')$, u' is a random number and t_3' is a timestamp. The proposed scheme provides user anonymity and data unlinkability, and thus exhibits untrackability [29]. Accordingly, the privacy of users is protected.

4.6. Resistance to Privileged Insider Attacks

Theorem 4. *The proposed scheme withstands privileged insider attacks.*

Proof of Theorem 4. In the registration phase, the user sends REG_i rather than (ID_i, PW_i) to *GWN*, where $REG_i = K_{UG} \oplus (ID^{pre}{}_i \| ID_i \| h(ID_i \| PW_i \| r_i)$, U_i's identity ID_i and password PW_i are protected by a random number r_i. Therefore, the privileged insider fails to obtain (ID_i, PW_i) and REG_i, and fails correctly to compute $TCR_i = D_1 \oplus h(ID_i \| PW_i \| r_i)$ (or $h(K_{GMN}||ID_i||E_i)$), so the proposed scheme withstands the privileged insider attack.

4.7. Resistance to Impersonation Attacks

Theorem 5. *The proposed scheme withstands impersonation attacks.*

Proof of Theorem 5. An adversary who tries to impersonate U_i fails to compute $TCR_i = D_1 \oplus h(ID_i \| PW_i \| r_i)$ and $X_3 = h(ID_i \| K_1 \| TCR_i \| t_3)$, and cannot send out the correct request messages $M_1 = \{DID_i, X_2, X_3, t_3\}$ in the login and authentication phase without the correct ID_i, PW_i and (D_1, r_i) in U_i's smart card, where t_3 is the timestamp. A failed login is detected by the *GWN* in Step 2 of the login and authentication phase, so the proposed scheme withstands impersonation attacks.

4.8. Resistance to Off-Line Password Guessing Attacks

Theorem 6. *The proposed scheme withstands off-line password guessing attacks.*

Proof of Theorem 6. In the proposed scheme, since reveal messages $M_1 = \{DID_i, X_2, X_3, t_3\}$, $M_2 = \{DID_G, X_2, Y_4, t_4\}$, $M_3 = \{Z_1, Z_2, Z_3, t_5\}$ and $M_4 = \{SID_j, Z_1, Z_3, t_5\}$ do not provide information about users' passwords PW_i, an adversary cannot confirm the accuracy of the passwords that have been guessed from M_1, M_2, M_3 and M_4, where $DID_i = ID_i \oplus K_1$, $K_1 = T_u(PK_G) \bmod p$, $X_2 = T_u(x) \bmod p$, $X_3 = h(ID_i \| K_1 \| TCR_i \| t_3)$ and $TCR_i = h(K_{GMN} || ID_i || E_i)$; $DID_G = ID_i' \oplus K_2$, $K_2 = h(Q_j \| t_4)$ and $Y_4 = h(Q_j \| ID_i' \| X_2 \| t_4)$; and $Z_1 = T_v(x) \bmod p$, $Z_2 = h(K_2' \| ID_i'' \| SID_j \| t_5)$, $Z_3 = h(sk \| ID_i'' \| SID_j \| t_5)$ and $sk = T_v(X_2) \bmod p$. Thus, off-line password guessing attacks are ineffective against the proposed scheme.

4.9. Resistance to Undetectable On-Line Password Guessing Attacks

Theorem 7. *The proposed scheme withstands on-line password guessing attacks.*

Proof of Theorem 7. Again, the revealed messages M_1, M_2, M_3 and M_4 do not provide information about a user's password PW_i. Accordingly, an attacker has difficulty in guessing the password in an on-line transaction, and the scheme thus resists undetectable on-line password guessing attacks.

4.10. Resistance to Stolen Verifier Attacks

Theorem 8. *The proposed scheme withstands stolen verifier attacks.*

Proof of Theorem 8. In the proposed scheme, the *GWN* keeps $(h(ID_i), E_i)$ in the verifier table for each user U_i. An adversary who steals the *GWN*'s verifier table and copies $(h(ID_i), E_i)$ still fails to compute $TCR_i = D_1 \oplus h(ID_i \| PW_i \| r_i)$, $DID_i = ID_i \oplus K_1$ and $X_3 = h(ID_i \| K_1 \| TCR_i \| t_3)$ without knowledge of user U_i's ID_i, PW_i, r_i and D_1, where u is a random number, $K_1 = T_u(PK_G) \bmod p$, $X_2 = T_u(x) \bmod p$ and t_3 is the timestamp. The adversary fails to send out $M_1 = \{DID_i, X_2, X_3, t_3\}$ in Step 1, and a failed login is detected by the *GWN*. Therefore, the proposed scheme resists stolen verifier attacks.

4.11. Resistance to Lost Smartcard Attacks

Theorem 9. *The proposed scheme withstands lost smartcard attacks.*

Proof of Theorem 9. An adversary who steals user U_i's smartcard and copies the message $(D_1, PK_G, T(.), x, p, h(.), r_i)$ still fails to compute $TCR_i = D_1 \oplus h(ID_i \| PW_i \| r_i)$ and $X_3 = h(ID_i \| K_1 \| TCR_i \| t_3)$, where t_3 is the timestamp, and so cannot send out the correct messages $M_1 = \{DID_i, X_2, X_3, t_3\}$ in Step 1 of the login and authentication phase without the correct ID_i and PW_i. The *GWN* will detect a failed login Step 2 of the login and authentication phase, so the proposed scheme withstands lost smartcard attacks.

4.12. Resistance to Many Logged-in Users Attacks

Theorem 10. *The proposed scheme withstands many-logged-in-users attacks.*

Proof of Theorem 10. Assume that U_i's login information $(ID_i, PW_i, T(.), x, p, h(.), r_i)$ is leaked to more than one non-registered user. The *GWN* also maintains a status-bit field and a last login field in its verifier table to prevent simultaneous duplicate logins. Therefore, the proposed scheme withstands many-logged-in-users attacks.

5. Performance Analyses and Functionality Comparisons

5.1. Performance Analyses

Table 2 compares the performance of the proposed scheme with those of the schemes developed by Yeh *et al*. [16], Xue *et al*. [8], Li *et al*. [9] and Kim *et al*. [35], where T_h is the execution time for a one-way hash operation; T_c is the execution time for a Chebyshev chaotic map operation, and T_e is the execution time for a scalar multiplication operation on an elliptic curve.

The first comparison made concerns the computational cost for user U_i, sensor node S_j and the gateway node *GWN*. The scheme of Yeh *et al*., [16] employs encryptions and decryptions on an elliptic curve, and has a greater computational cost than related schemes [8,9,35], which use only hash operations. Since T_c approximates T_h, where T_h is obtained by using the hash functions SHA-1 and MD5 [36–38], the proposed scheme requires six chaotic map operations and 13 hash function operations and so has a low computational burden.

Table 2. The performance comparisons of the related schemes and the proposed scheme.

		Yeh *et al*. [16]	Xue *et al.* [8]	Li *et al.* [9]	Kim *et al.* [35]	Our Scheme
Computations	U_i	2 T_e + 1 T_h	7 T_h	9 T_h	8 T_h	3 T_c + 3 T_h
	S_j	2 T_e + 3 T_h	5 T_h	6 T_h	2 T_h	2 T_c + 4 T_h
	GWN	4 T_e + 4 T_h	10 T_h	11 T_h	8 T_h	1 T_c + 6 T_h
	Total	8 T_e + 8 T_h	22 T_h	26 T_h	18 T_h	6 T_c + 13 T_h

5.2. Functionality Comparisons

Table 3 compares the proposed scheme and related schemes in terms of functionality, and specifically the meeting of security requirements and resistance to possible attacks. The schemes that were developed by Yeh *et al.*, Xue *et al.*, Li *et al.* and Kim *et al.* all fail to protect users' privacy. Additionally, the scheme of Yeh *et al.* fails to withstand password guessing, lost smart card and many-logged-in-users attacks. The scheme of Xue *et al.* fails to withstand privileged insider, password guessing, stolen verifier, lost smart card and many-logged-in-users attacks. The scheme of Li *et al.* fails to withstand impersonation and stolen verifier attacks. Only the proposed scheme withstands all possible attacks and protects privacy. Thus, the proposed scheme provides greater functionality; exhibits more favorable security-related properties, and has a lower computational cost than the other schemes.

Table 3. The functionality comparisons of the related schemes and the proposed scheme.

	Yeh *et al.* [16]	Xue *et al.* [8]	Li *et al.* [9]	Kim *et al.* [35]	Our Scheme
Providing mutual authentication	Yes	Yes	Yes	Yes	Yes
Providing session key security	Yes	Yes	Yes	Yes	Yes
Providing privacy protection	No	No	No	No	Yes
Resisting privileged insider attacks	Yes	No	Yes	Yes	Yes
Resisting to impersonation attacks	Yes	Yes	No	Yes	Yes
Resisting password guessing attacks	No	No	Yes	Yes	Yes
Resisting stolen verifier attacks	Yes	No	No	Yes	Yes
Resisting lost smartcard attacks	No	No	Yes	Yes	Yes
Resisting many logged-in users attacks	No	No	Yes	Yes	Yes

6. Conclusions

This study addresses the weaknesses of the temporal credential-based authenticated key agreement scheme developed by Li *et al.*, which enables an adversary to impersonate legitimate users, to perform a stolen verifier attack to calculate all used session keys and transmitted secrets of users and sensor nodes, and to reveal users' identities. A new temporal credential-based authenticated key agreement scheme that uses chaotic maps is developed for WSNs. The proposed scheme protects each user's identity using a temporary secret key; conceals each user's private parameters, and reduces the number of redundant parameters concerning the user's identity and password in the verifier table of the *GWN*. Therefore, the proposed scheme does not have any of the weaknesses of previous schemes. Additionally, session key security is based on the extended chaotic maps-based Diffie-Hellman problem, and the proposed scheme thus exhibits perfect forward secrecy and known-key security. The proposed scheme not only eliminates the weaknesses of previous approaches, but also increases security and efficiency.

Acknowledgments

This research was supported by Ministry of Science and Technology under the grants MOST 103-2221-E-320 -003 and TCRPP103008. Ted Knoy is appreciated for his editorial assistance.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Delgado-Mohatar, O.; Fuster-Sabater, A.; Sierra, J.M. A light-weight authentication scheme for wireless sensor networks. *Ad Hoc Netw.* **2011**, *9*, 727–735.
2. Li, C.T.; Hwang, M.S. A lightweight anonymous routing protocol without public key en/decryptions for wireless *ad hoc* networks. *Inf. Sci.* **2011**, *181*, 5333–5347.
3. Li, Z.; Gong, G. Computationally efficient mutual entity authentication in wireless sensor networks. *Ad Hoc Netw.* **2011**, *9*, 204–215.
4. Das, A.K. Improving Identity-based Random Key Establishment Scheme for Large-scale hierarchical wireless sensor networks. *Int. J. Netw. Secur.* **2012**, *14*, 1–21.
5. Mi, Q.; Stankovic, J.A.; Stoleru, R. Practical and secure localization and key distribution for wireless sensor networks. *Ad Hoc Netw.* **2012**, *10*, 946–961.
6. Jie, H.; Guohua, O. A public key polynomial-based key pre-distribution scheme for large-scale wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* **2012**, *16*, 45–64.
7. Han, K.; Kim, K.; Choi, W.; Choi, H.H.; Seo, J.; Shon, T. Efficient authenticated key agreement protocols for dynamic wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* **2012**, *14*, 251–269.
8. Xue, K.; Ma, C.; Hong, P.; Ding, R. A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. *J. Netw. Comput. Appl.* **2013**, *36*, 316–323.

9. Li, C.T.; Weng, C.Y.; Lee, C.C. An advanced temporal credential-based security scheme with mutual authentication and key agreement for wireless sensor networks. *Sensors* **2013**, *13*, 9589–9603.
10. Wong, K.H.M.; Zheng, Y.; Cao, J.; Wang, S. A dynamic user authentication scheme for wireless sensor networks. In Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, Taichung, Taiwan, 5–7 June 2007; pp. 32–58.
11. Xu, J.; Zhu, W.; Feng, D. An improved smart card based password authentication scheme with provable security. *Comput. Stand. Interfaces* **2009**, *31*, 723–728.
12. Das, M.L. Two-factor user authentication scheme in wireless sensor networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1086–1090.
13. He, D.; Gao, Y.; Chan, S.; Chen, C.; Bu, J. An enhanced two-factor user authentication scheme in wireless sensor networks. *Ad Hoc Sens. Wirel. Netw.* **2010**, *10*, 361–371.
14. Khan, M.K.; Alghathbar, K. Cryptanalysis and security improvements of two-factor user authentication in wireless sensor networks. *Sensors* **2010**, *10*, 2450–2459.
15. Song, R. Advanced smart card based password authentication protocol. *Comput. Stand. Interfaces* **2010**, *32*, 321–325.
16. Yeh, H.L.; Chen, T.H.; Liu, P.C.; Kim, T.H.; Wei, H.W. A secure authentication protocol for wireless sensor networks using elliptic curves cryptography. *Sens. J.* **2011**, *11*, 4767–4779.
17. Chen, T.H.; Shih, W.K. A robust mutual authentication protocol for wireless sensor networks. *ETRI J.* **2010**, *32*, 704–712.
18. Bergamo, P.; D'Arco, P.; Santis, A.; Kocarev, L. Security of public-key cryptosystems based on Chebyshev polynomials. *IEEE Trans. Circuits Syst. I* **2005**, *52*, 1382–1393.
19. Kocarev, L.; Tasev, Z. Public-key encryption based on Chebyshev maps. In Proceedings of the International Symposium on Circuits and Systems, Bangkok, Thailand, 25–28 May 2003; pp. III-28–III-31.
20. Mason, J.C.; Handscomb, D.C. *Chebyshev. Polynomials*; Chapman & Hall/CRC, Boca Raton, Florida, FL, USA, 2003.
21. Zhang, L. Cryptanalysis of the public key encryption based on multiple chaotic systems. *Chaos Solitons. Fractals* **2008**, *37*, 669–674.
22. Lee, C.C.; Chen, C.L.; Wu, C.Y.; Huang, S.Y. An extended chaotic maps-based key agreement protocol with user anonymity. *Nonlinear Dyn.* **2012**, *69*, 79–87.
23. Lee, C.C.; Hsu, C.W. A secure biometric-based remote user authentication with key agreement scheme using extended chaotic maps. *Nonlinear Dyn.* **2013**, *71*, 201–211.
24. Lee, T.F. An efficient chaotic maps-based authentication and key agreement scheme using smartcards for telecare medicine information systems. *J. Med. Syst.* **2013**, *37*, doi:10.1007/s10916-013-9985-9.
25. Lee, T.F. Verifier-based three-party authentication schemes using extended chaotic maps for data exchange in telecare medicine information systems. *Comput. Meth. Programs Biomed.* **2014**, *117*, 464–472.
26. Farash, M.S.; Attari, M.A. An efficient and provably secure three-party password-based authenticated key exchange protocol based on Chebyshev chaotic maps. *Nonlinear Dyn.* **2014**, *77*, 399–411.

27. Lou, D.C.; Lee, T.F.; Lin, T.H. Efficient biometric authenticated key agreements based on extended chaotic maps for telecare medicine information systems. *J. Med. Syst*. **2015**, *39*, doi:10.1007/s10916-015-0240-4.
28. Stallings, W. *Cryptography and Network Security: Principles and Practice*, 5th ed.; Pearson: Upper Saddle River, NJ, USA, 2011.
29. Lee, T.F. User authentication scheme with anonymity, unlinkability and untrackability for global mobility networks. *Secur. Commun. Netw*. **2013**, *6*, 1404–1413.
30. Abdalla, M.; Pointcheval, D. Simple password-based authenticated key protocols. Proc. Topics in Cryptology—CT-RSA 2005, San Francisco, CA, USA, 14–18 February 2005; pp. 191–208.
31. Bellare, M.; Pointcheval, D.; Rogaway, P. Authenticated key exchange secure against dictionary attacks. Proc. Adv. Cryptol. Eurocrypt 2000, Bruges, Belgium, 14–18 May 2000; pp. 122–138.
32. Boyko, V.; MacKenzie, P.; Patel, S. Provably secure password-based authenticated key exchange protocols using Diffie-Hellman. Proc. Adv. Cryptol. Eurocrypt 2000, Bruges, Belgium, 14–18 May 2000; pp.156–171.
33. Lee, T.F.; Hwang, T. Provably secure and efficient authentication techniques for the global mobility network. *J. Syst. Soft.* **2011**, *84*, 1717–1725.
34. Shoup, V. Sequences of Games: A Tool for Taming Complexity in Security Proofs, Manuscript. Available online: http:// www.shoup.net (accessed on 18 January 2015).
35. Kim, J.; Lee, D.; Jeon, W.; Lee, Y.; Won, D. Security analysis and improvements of two-factor mutual authentication with key agreement in wireless sensor networks. *Sensors* **2014**, *14*, 6443–6462.
36. Xiao, D.; Liao, X.; Deng, S. One-way hash function construction based on the chaotic map with changeable-parameter. *Chaos Solitons Fractals* **2005**, *24*, 65–71.
37. Wu, S.; Chen, K. An efficient key-management scheme for hierarchical access control in E-Medicine system. *J. Med. Syst*. **2012**, *36*, 2325–2337.
38. Cheng, Z.Y.; Liu, Y.; Chang, C.C.; Chang, S.C. Authenticated RFID security mechanism based on chaotic maps. *Secur. Commun. Netw*. **2013**, *6*, 247–256.