Peer-to-Peer Netw. Appl. (2016) 9:223–244
DOI 10.1007/s12083-014-0324-9

A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks

Ashok Kumar Das

Received: 8 July 2014 / Accepted: 3 December 2014 / Published online: 23 December 2014

Abstract User authentication is one of the most important security services required for the resource-constrained wireless sensor networks (WSNs). In user authentication, for critical applications of WSNs, a legitimate user is allowed to query and collect the real-time data at any time from a sensor node of the network as and when he/she demands for it. In order to get the real-time information from the nodes, the user needs to be first authenticated by the nodes as well as the gateway node (GWN) of WSN so that illegal access to nodes do not happen in the network. Recently, Jiang et al. proposed an efficient two-factor user authentication scheme with unlinkability property in WSNs Jiang (2014). In this paper, we analyze Jiang et al.'s scheme. Unfortunately, we point out that Jiang et al.'s scheme has still several drawbacks such as (1) it fails to protect privileged insider attack, (2) inefficient registration phase for the sensor nodes, (3) it fails to provide proper authentication in login and authentication phase, (4) it fails to update properly the new changed password of a user in the password update phase, (5) it lacks of supporting dynamic sensor node addition after initial deployment of nodes in the network, and (6) it lacks the formal security verification. In order to withstand these pitfalls found in Jiang et al.'s scheme, we aim to propose a three-factor user authentication scheme for WSNs. Our scheme preserves the original merits of Jiang et al.'s scheme. Our scheme is efficient as compared to Jiang et al.'s scheme and other schemes. Furthermore, our scheme provides better security features and higher security level than other schemes. In addition, we simulate our scheme for the formal security analysis using the widely-accepted AVISPA (Automated Validation of Internet Security Protocols and Applications) tool. The simulation results clearly demonstrate that our scheme is also secure.

A. K. Das (✉)
Center for Security, Theory and Algorithmic Research,
International Institute of Information Technology,
Hyderabad 500 032, India
e-mail: iitkgp.akdas@gmail.com; ashok.das@iiit.ac.in.

Keywords Wireless sensor networks · Authentication · Fuzzy extractor · Biometrics · Password · Smart cards · User anonymity · Unlinkability · Security

1 Introduction

A wireless sensor network (WSN) is composed of several tiny computing nodes, called the sensor nodes or motes, which could be order of hundreds or several hundreds. Typically the sensor nodes are deployed randomly in a target field (also called the deployment area) for the purpose of sensing important information from their surrounding fields and then sending those sensing information to the nearby base station or the gateway node (GWN). Sensor nodes communicate among each other by short range radio communications. The base station is a computationally well-equipped node in the network, whereas the sensor nodes are resource-starved. Since the GWN can reach all the sensor nodes in a network, depending on the applications, the GWN can be located either in the center or at a corner of the network.

Sensor networks are widely deployed in a variety of applications ranging from military to environmental and medical research. In many applications, such as target tracking, battlefield surveillance and intruder detection, WSNs often operate in hostile and unattended environments. Therefore, there is a strong need for protecting the sensing data and sensing readings. In wireless environments, an adversary not only can eavesdrop the radio traffic, but also has the ability to intercept or interrupt the exchanged messages. Thus, many protocols and algorithms do not work in hostile environments without adequate security measures. Hence, security becomes one of the major concerns when there are potential attacks against sensor networks.

Consider the scenario of battle field surveillance which is one of the major military applications. A large number of sensor nodes are rapidly deployed in a battlefield via airplanes or trucks. Each individual sensor node monitors conditions and activities in its surroundings after deployment in the battle field and then reports these sensing observations to the GWN via wireless communications through its neighboring sensor nodes. The GWN then can conduct a more accurate detection on the activities (for example, possible attacks) of the opposing force after collecting a large number of sensing observations from the sensor nodes. Thus, the appropriate decisions as well as responses can be made quickly in the battle field.

In user authentication in WSN, a legitimate user is allowed to query and collect the real-time data at any time from a sensor node of the network as and when he/she demands for it. As most of the WSN applications are real-time based, so the users (called the external parties) are generally interested in accessing the real-time information. This could happen if we allow the users to access the real-time data directly from the nodes inside WSN. Usually, the information from nodes are gathered periodically in the GWN and hence, the gathered information may not be real-time. In order to access the real-time information from the nodes, the user needs to be first authenticated to the sensor nodes as well as the GWN so that illegal access to nodes do not happen. As a result, the user authentication problem becomes a very important topic in research of WSN security.

Several two-factor user authentication schemes have been proposed in the literature [6, 13, 17, 18, 20, 22, 28, 29, 35, 37, 40, 41], which use a user's smart card and password. However, most of these proposals are insecure against different known attacks. Recently, Jiang et al. [21] proposed an efficient two-factor user authentication scheme for WSNs. However, in this paper, we analyze the recently proposed Jiang et al.'s scheme [21] and show that unfortunately their scheme has several drawbacks. In order to remedy these drawbacks found in Jiang et al.'s scheme, we aim to propose a secure and robust temporal credential-based user anonymity and unlinkability-preserving three factor user authentication scheme for wireless sensor networks. In our scheme, we use a user's personal biometric as the third factor apart from that user's smart card and password.

Tan [36] extended the security requirements of two-factor authentication schemes to three-factor authentication schemes, which are given below:

- *Mutual authentication.* After run of the protocol, the server should believe that the remote user is a legitimate registered client. The user also believes that the communicating party is the server which the user intended to login to.
- *Server not knowing password and biometric.* The registration center (server) should not have any information about the registered user's password and personal biometrics. This is extremely required because several users may apply the same password to access different servers in the real applications. As a result, if a privileged insider of the registration center knows the password or biometrics of a user U_i, he/she may impersonate U_i for accessing the services from other servers.
- *Freedom of password and biometric update.* A user should be allowed to change/update freely his/her password as well as biometric template without contacting the server. The server must be totally unaware of the change of the user's password and biometric template.
- *Three-factor security.* In the security model for three-factor authentication schemes, an adversary can have full control over the communication channel between the users and the server during the login phase and the authentication and key agreement phase. In the three-factor security adversary model, the adversary is modeled to have at most two of the following three abilities, but it is not allowed to have all the three abilities. The adversary can use the techniques in [24, 27] to extract the information from the smart card, obtain the password, or access the biometric template. This model is not applicable for a privileged insider attack, where a legal user U is himself/herself is an insider attacker, who knows all these three factors: smart card, password and personal biometrics.

1.1 Related work

Watro et al. [38] proposed a public-key cryptography based user authentication scheme in WSNs, which is known as TinyPK. Their scheme is based on RSA cryptosystem [31] and Diffie-Hellman key exchange protocol [14]. However, their protocol is insecure as pointed out in [13]. Later, Wong et al. [39] proposed a user authentication scheme, which is password-based and uses the hash function. Their scheme is vulnerable to many logged-in users with the same login id attack and also suffers from the stolen-verifier attack, because both the GWN and login-sensor node need to maintain a lookup table of the registered users' credentials. M. L. Das [13] introduced an efficient scheme, which uses password and smart card of a user. This scheme is however vulnerable to the denial-of-service attack as well as node capture attack [12]. Many researchers then inspired from this work and proposed several improvements [6, 17, 18, 20, 22, 28, 37].

Fan et al. [17] proposed a user authentication scheme based on two-tiered WSNs. Their scheme is efficient and also resists the denial-of-service attack. Chen and Shih [6] proposed a robust mutual authentication scheme for WSNs, which withstands te security pitfalls of M. L. Das's scheme [12]. Yuan et al. [42] proposed a biometric-based user authentication scheme, which uses the similar concept of M. L. Das's scheme [13]. Their scheme has the same drawbacks as found in [13]. Das et al. [9, 12] proposed a two-factor dynamic password-based user authentication scheme for hierarchical wireless sensor networks, which has several attractive features such as dynamic node addition, correct password change phase and session key agreement.

Yoo et al. [41] proposed a robust two-factor user authentication scheme. But their scheme does not provide user privacy protection [21]. Sun et al. [35] proposed an improvement over Khan-Alghathar's scheme [22] in order to withstand the GWN impersonation attack. As pointed out in [21], Sun et al.'s scheme [35] does not provide user privacy protection, mutual authentication between a user and the GWN, and also session key agreement between a user and an accessed sensor node. Xue et al. [40] further proposed a temporal credential-based mutual authentication and key agreement scheme for WSNs, which is efficient due to use of the hash function. However, Jiang et al. [21] pointed out that Xue et al.'s scheme is vulnerable to identity guessing attack, tracking attack, privileged-insider attack and stolen smart card attack. To remedy these weaknesses, Jiang et al. further proposed a two-factor user authentication scheme for WSNs. However, in this paper, we again analyze the recently proposed Jiang et al.'s scheme [21] and show that unfortunately their scheme has several drawbacks, which are discussed in detail in Section 4.

1.2 Threat model

We make use of the Dolev-Yao threat model [16] in which two communicating parties communicate over an insecure channel. Any adversary (attacker or intruder) can eavesdrop the transmitted messages over the public insecure channel and he/she has the ability to modify, delete or change the contents of the transmitted messages. Usually the smart card of a user is equipped with the tamper-resistant hardware. However, if a user's smart card is stolen or lost, an attacker can still know all the sensitive information stored in its memory by monitoring the power consumption of the smart card [24, 27].

1.3 Notations

In this paper, we use the notations listed in Table 1 throughout the paper in order to analyze Jiang et al.'s scheme, and also describe and analyze our proposed scheme.

1.4 Our contributions

We list the following contributions made in this paper:

- We first analyze the security of the recently proposed Jiang et al.'s scheme [21], and find that their scheme has several drawbacks.
- In order to remedy, the drawbacks found in Jiang et al.'s scheme, we propose a secure and robust temporal credential based user anonymity and unlinkability preserving three-factor user authentication scheme in WSNs. Our scheme makes use of three factors, namely user's identity, password and biometric.
- Through the rigorous informal and formal security analysis, we show that our scheme can withstands several known attacks including the attacks found in Jiang et al.'s scheme.
- We analyze the efficiency of our proposed scheme, and show that our scheme is also efficient as compared to Jiang et al.'s scheme and other related schemes in WSNs.
- We simulate our scheme for the formal security analysis using the widely-accepted AVISPA (Automated Validation of Internet Security Protocols and Applications) tool [2] and the simulation results clearly demonstrate that our scheme is secure.

1.5 Roadmap of the paper

The remainder of this paper is organized as follows. In Section 2, we discuss some basic preliminaries such as one-way hash function and fuzzy extractor technique, which are essential for better understanding the paper. In Section 3, we review the recently proposed Jiang et al.'s scheme [21]. We then analyze the security of their scheme in Section 4. In Section 5, we propose a secure and robust three-factor user authentication scheme in WSNs. In Section 6, through the formal and informal security analysis we show that our scheme has the ability to tolerate various known attacks. In Section 7, we simulate our scheme for the formal security verification using AVISPA tool. We compare the performance of our scheme with Jiang et al.'s scheme and other schemes in Section 8. Finally, we conclude the paper in Section 9.

2 Mathematical preliminaries

In this section, we briefly describe some mathematical preliminaries, which are essential for describing and analyzing Jiang et al.'s scheme [21] as well as our proposed scheme.

2.1 Collision-resistant one-way hash function

We define the formal definition of a one-way collision-resistant hash function as follows ([10, 32, 34]).

Definition 1 A collision-resistant one-way hash function $h : A \to B$, where $A = \{0, 1\}^*$ and $B = \{0, 1\}^n$, is a deterministic algorithm that takes an input as an arbitrary length binary string $x \in A$ and produces an output $y \in B$, a binary string of fixed-length, n. Let $Adv_{\mathcal{A}}^{HASH}(t_1)$ denote an adversary (attacker) $\mathcal{A}$'s advantage in finding collision. Then, we have

$$Adv_{\mathcal{A}}^{HASH}(t_1) = Pr[(x, x') \in_R \mathcal{A} : x \neq x', \text{ and } h(x) = h(x')],$$

where $Pr[E]$ denotes the probability of a random event E, and $(x, x') \in_R \mathcal{A}$ denotes the pair (x, x') is selected randomly by $\mathcal{A}$. In this case, the adversary $\mathcal{A}$ is also allowed to be probabilistic and the probability in the advantage is computed over the random choices made by the adversary $\mathcal{A}$ with the execution time t_1. The hash function $h(\cdot)$ is then called collision-resistant, if $Adv_{\mathcal{A}}^{HASH}(t_1) \leq \epsilon_1$, for any sufficiently small $\epsilon_1 > 0$.

2.2 Fuzzy extractor

We briefly describe the extraction process of key data from the given biometric of a user using a fuzzy extractor technique. The output of a conventional hash function $h(\cdot)$ is sensitive and it may also return completely different outputs even if there is a little variation in inputs. Note that the biometric information is prone to various noises during data acquisition, and the reproduction of actual biometric is hard in common practice. To avoid such problem, a fuzzy extractor method [5, 15, 19] is preferred, which can extract a uniformly random string and a public information from the biometric template with a given error tolerance t. In the reproduction process, the fuzzy extractor recovers the original biometric key data for a noisy biometric using public information and t. Let $\mathcal{M} = \{0, 1\}^v$ denote a finite v-dimensional metric space of biometric data points, $d : \mathcal{M} \times \mathcal{M} \to \mathbb{Z}^+$ a distance function, which can be used to calculate the distance between two points based on the metric chosen, l the number of bits of the output string, and t the error tolerance, where $\mathbb{Z}^+$ represents the set of all positive integers.

Definition 2 The fuzzy extractor is a tuple $(\mathcal{M}, l, t)$, which is composed of the following two algorithms, called *Gen* and *Rep*:

- **Gen:** It is a probabilistic algorithm, which takes a biometric information $B_i \in \mathcal{M}$ as input, and then outputs a secret key data $\sigma_i \in \{0, 1\}^l$ and a public reproduction parameter τ_i, where $Gen(B_i) = \{\sigma_i, \tau_i\}$.
- **Rep:** This is a deterministic algorithm, which takes a noisy biometric information $B_i' \in \mathcal{M}$ and a public parameter τ_i and t related to B_i, and then it reproduces (recovers) the biometric key data σ_i. In other words, we have $Rep(B_i', \tau_i) = \sigma_i$ provided that the condition $d(B_i, B_i') \leq t$ is met.

One can refer to [5, 15] for more detailed description of the fuzzy extractor and the extraction procedure.

3 Review of Jiang et al.'s scheme

In this section, we review in brief the recently proposed Jiang et al.'s scheme [21]. We use the notations given in Table 1 for describing and analyzing Jiang et al.'s scheme.

3.1 Registration phase

In this phase, a legal user U_i registers or re-registers with the GWN. In order to register to the GWN, the user U_i need to execute the following steps:

Step 1. U_i first selects a unique identity ID_i and a chosen password PW_i. After that U_i generates a random value r and computes $RPW_i = h(r||PW_i)$ and sends the registration request message $R = \langle ID_i, RPW_i \rangle$ to the GWN via a secure channel.

Step 2. After receiving the registration request message $R = \langle ID_i, RPW_i \rangle$ from the user U_i, the GWN verifies ID_i and rejects this request if ID_i is invalid. Otherwise, the GWN computes the temporal credential $TC_i = h\ (K_{GWN-U}||ID_i||TE_i)$ and $PTC_i = TC_i \oplus RPW_i$ for the user U_i. The GWN then initializes the temporary identity TID_i of the user U_i and stores the tuple (TID_i, ID_i, TE_i) in its verification table. The GWN issues a smart card containing to the information $\{h(\cdot), TID_i, TE_i, PTC_i\}$ to U_i and sends it via a secure channel.

Table 1 Notations used in this paper

Symbol	Description
GWN	WSN gateway node (base station)
U_i	i^{th} user
SC_i	Smart card of U_i
ID_i	Identity of user U_i
PW_i	Password of user U_i
B_i	Biometric information of U_i
K	1024-bit secret number known to U_i only
$h(\cdot)$	Secure collision-free one-way hash function
X_s	1024-bit secret master key of S_j
SN_j	j^{th} sensor node in WSN
ID_{SN_j}	Identity of SN_j
TE_i	Expiration time of U_i's temporal credential
TS_X	Current timestamp of an entity X
$Gen(\cdot)$	Fuzzy generator function
$Rep(\cdot)$	Fuzzy reproduction function
t	Error tolerance threshold used in fuzzy extractor
ΔT	Maximum transmission delay
$A \oplus B$	Bitwise XORed of data A with data B
$A\|\|B$	Data A concatenates with data B

Step 3. After receiving the smart card, U_i stores the random number r in the smart card.

The registration phase for all the deployed sensor nodes is as follows:

Step 1. A sensor node SN_j submits its identifier ID_{SN_j} to the GWN through a secure channel.

Step 2. Upon receiving the message, the GWN computes the temporal credential for SN_j as $TC_j = h(K_{GWN-S} || ID_{SN_j})$, where K_{GWN-S} is the GWN's private key only known to the GWN. The GWN then sends the message $\langle TC_j \rangle$ to SN_j.

Step 3. When the sensor node SN_j receives the message in Step 2, SN_j stores TC_j as its temporal credential.

3.2 Login and authentication phase

In this phase, in order to login to the GWN and authenticate by the sensor nodes in WSN, the following steps are executed by the user U_i, the GWN and the login-sensor node SN_j:

Step 1. U_i first inserts his/her smart card to a terminal and then inputs his/her identity ID_i and password PW_i. The smart card then generates a current timestamp TS_4 and also a random key K_i. After that the smart card computes $TC_i = PTC_i \oplus h(r||PW_i)$, $PKS_i = K_i \oplus h(TC_i\,||TS_4)$, $C_i = h(ID_i||K_i||TC_i||TS_4)$. U_i then sends the login request message $\langle TID_i, C_i, PKS_i, TS_4 \rangle$ to the GWN via a public channel.

Step 2. After receiving the message in Step 1, the GWN checks the condition $|T^*_{GWN} - TS_4| < \Delta T$, where T^*_{GWN} is the current system timestamp of the GWN and ΔT the maximum transmission delay. If this condition does not hold, the GWN immediately terminates this phase and sends the reject message REJ back to the user U_i. Otherwise, the GWN fetches the identity ID_i corresponding to TID_i in the verification table, and computes $TC_i = h(K_{GWN-U}||ID_i||TE_i)$ and $C^*_i = h(ID_i||K_i||TC_i||TS_4)$. If $C^*_i \neq C_i$, the GWN sends reject message REJ back to the user U_i and terminates this phase. Otherwise, the GWN authenticates the user U_i and further, computes $K_i = PKS_i \oplus h(TC_i||TS_4)$. In addition, the GWN computes the accessed sensor node SN_j's temporal credential $TC_j = h(K_{GWN-S}||ID_{SN_j})$, $C_{GWN} = h(TID_i||TC_j||TS_5)$ and $PKS_{GWN} = K_i \oplus h(TC_j||TS_5)$, where TS_5 is the current timestamp of the GWN. The GWN then sends the message $\langle TS_5, TID_i, C_{GWN}, PKS_{GWN} \rangle$ to the sensor node SN_j.

Step 3. After receiving the message in Step 2, the sensor node SN_j checks the condition $|T^*_j - TS_5| < \Delta T$, where T^*_j is the current system timestamp of SN_j. If this condition fails, SN_j terminates the current session. Otherwise, SN_j computes $C^*_{GWN} = h(TID_i||TC_j||TS_5)$. If $C^*_{GWN} \neq C_{GWN}$, the sensor node SN_j rejects this message. Otherwise, SN_j confirms that the sender of the received message sent in Step 2 is an authorized GWN. After that SN_j computes $K_i = PKS_{GWN} \oplus h(TC_j||TS_5)$. SN_j generates the current timestamp TS_6 and also a random key K_j, and computes $C_j = h(K_j||\,TID_i||ID_{SN_j}||TS_6)$, $PKS_j = K_j \oplus h(K_i||TS_6)$. SN_j finally sends the message $\langle ID_{SN_j}, TS_6, C_j, PKS_j \rangle$ to the GWN.

Step 4. If the timeliness of TS_6 in the received message $\langle ID_{SN_j}, TS_6, C_j, PKS_j \rangle$ is verified successfully, the GWN computes $C^*_j = h(K_j||ID_i||ID_{SN_j}||TS_6)$ and checks if $C^*_j = C_j$. If it is valid, SN_j is considered as a legitimate sensor node. The GWN then proceeds to generate a new temporary identity TID'_i and computes $D_{GWN} = TID'_i \oplus h(K_i||TS_7)$, where TS_7 is the current timestamp of the GWN. The GWN replaces TID_i with TID'_i in its verification table, and computes $E_{GWN} = h(ID_i||ID_{SN_j}||TC_i||D_{GWN}||K_j||TS_7)$. The GWN finally sends the message $\langle ID_{SN_j}, TS_7, PKS_j, D_{GWN}, E_{GWN} \rangle$ to the user U_i.

Step 5. Once the timeliness of TS_7 for the received message $\langle ID_{SN_j}, TS_7, PKS_j, D_{GWN}, E_{GWN} \rangle$ is verified by the user U_i, U_i proceeds to compute $TID'_i = D_{GWN} \oplus h(K_i||TS_7)$, $K_j = PKS_j \oplus h(K_i||TS_6)$ and E^*_{GWN}

$= h(ID_i||ID_{SN_j}||Tc_i||D_{GWN}||K_j||TS_7)$. If the condition $E^*_{GWN} = E_{GWN}$, the user U_i confirms that both SN_j and the GWN are legitimate. After that U_i replaces TID_i with TID'_i in the memory of the smart card, and computes the shared session key with the sensor node SN_j as $SK_{ij} = h(K_i \oplus K_j)$. The sensor node SN_j also computes the same shared session key with the user U_i as $SK_{ij} = h(K_i \oplus K_j)$. Finally, both U_i and SN_j can use SK_{ij} for their future secure communications.

3.3 Password update phase

In this phase, if a legal user U_i wants to change his/her current password, the U_i needs to insert his/her smart card to a terminal and then input his/her identity ID_i, old password PW_i and a new password PW'_i. After that the smart card computes $PTC'_i = TC_i \oplus RPW_i \oplus h(r||PW'_i)$ and replaces PTC_i with PTC'_i.

4 Drawbacks of Jiang et al.'s scheme

In this section, we analyze the security of Jiang et al.'s scheme. We find that their scheme has the following drawbacks.

4.1 Privileged insider attack

During the registration phase of Jiang et al.'s scheme, a legal user U_i submits the registration request message $R = \langle ID_i, RPW_i\rangle$ to the GWN via a secure channel, where r is a random number generated by the user U_i and $RPW_i = h(r||PW_i)$. After issuing the smart card containing the information $\{h(\cdot), TID_i, TE_i, PTC_i\}$ by the GWN, the user U_i stores the secret number r in his/her smart card. In this attack, we assume that an insider of the GWN attains the lost/stolen smart card of the legal user U_i. After that the insider being an attacker can mount the offline password guessing attack using dictionary attack in order to derive a low-entropy password PW_i of the user U_i using the following steps:

Step 1. Extract the information stored in the lost/stolen smart card of the user U_i using the power analysis attack [24, 27] (described in our threat model in Section 1.2). Thus, the insider knows r. Furthermore, note that the insider has $RPW_i = h(r||PW_i)$.

Step 2. Pick a guessed password PW^*_i and compute $RPW^*_i = h(r||PW^*_i)$.

Step 3. Check if $RPW^*_i = RPW_i$. If there is a match, the insider is successful in finding the correct password PW_i of the user U_i and terminates the procedure. Otherwise, the insider discards this guessed password and guesses a new password, and goes to Step 2.

It is thus clear that an insider of the GWN is successful in deriving the correct password PW_i of a legal user U_i in a relatively small dictionary. Hence, Jiang et al.'s scheme fails to protect the privileged insider attack.

4.2 Inefficient registration phase

In Jiang et al.'s scheme, the registration phase for all the deployed sensor nodes is performed online after the sensor nodes are deployed in a target field. Usually, due to large number of sensor nodes deployment in WSN, it is not preferable to have a procedure where each sensor node can register online with the WSN. Rather, in most of the WSN applications, this is performed in offline by the GWN. Note that in Jiang et al.'s scheme, a deployed sensor node SN_j sends its identity ID_{SN_j} to the GWN through a secure channel. This means that each sensor node SN_j needs to establish a secret key with the GWN. After receiving the message the GWN computes the temporal credential for SN_j as $TC_j = h(K_{GWN-S}||ID_{SN_j})$ and sends TC_j to SN_j. SN_j then stores TC_j for authentication purpose later. It is then clear that this procedure involves a lot of communication and computational overheads due to large number of sensor nodes involved during this process. Thus, Jiang et al.'s is inefficient for this registration phase for all sensor nodes in WSN.

4.3 Fails to provide proper authentication in login and authentication phase

This analysis is similar to that presented in [8]. In practice, a user U_i is expected to keep different passwords for different applications for security reasons. In this case, we assume that U_i enters his/her password PW_i incorrectly by mistake during the login and authentication phase of Jiang et al.'s scheme. Let the entered wrong password be PW'_i, where $PW'_i \neq PW_i$. Then, in Step 1 of the login and authentication phase, the smart card of the user U_i computes

$$\begin{aligned} TC_i &= PTC_i \oplus h(r||PW'_i) \\ &\neq PTC_i \oplus h(r||PW_i) \\ &\neq h(K_{GWN-U}||ID_i||TE_i). \end{aligned}$$

After that the smart card of U_i computes $PKS_i = K_i \oplus h(TC_i||TS_4)$, $C_i = h(ID_i||K_i||TC_i||TS_4)$, and sends the login request message $\langle TID_i, C_i, PKS_i, TS_4\rangle$ to the GWN. After receiving this login request message, the GWN checks the timeliness of TS_4, and if it is valid, the GWN fetches ID_i corresponding to TID_i in its verification table. Note that the GWN computes $TC_i = h(K_{GWN-U}||ID_i||TE_i)$ and $C^*_i = h(ID_i||K_i||TC_i||TS_4)$.

When the GWN verifies the condition $C_i^* = C_i$, it will fail because of incorrect password PW_i' ($\neq PW_i$) entered by the user U_i. However, in this case, U_i is totally unaware of the fact that he/she has entered password incorrectly. On the other hand, U_i may think the GWN as a cheater. Thus, Jiang et al.'s scheme fails to provide strong authentication during the login and authentication phase, because their scheme does not provide any mechanism for password verification by the smart card before the login request message is sent to the GWN.

4.4 Fails to update new password in password update phase

Similar to the analysis in Section 4.3, we also assume that a legal user U_i wants to change his/her password, and enters his/her old password PW_i^* incorrectly by mistake, where $PW_i \neq PW_i^*$. After that U_i enters his/her new changed password PW_i'. Note that during the password update phase of Jiang et al.'s scheme, the smart card of U_i never verifies whether the entered old password is correct. Due this problem, when the smart card computes $PTC_i' = TC_i \oplus RPW_i \oplus h(r||PW_i')$, where $RPW_i = h(r||PW_i^*) \neq h(r||PW_i)$, then $PTC_i' \neq TC_i \oplus h(r||PW_i')$. It is also noticed that there is a design flaw in computing PTC_i' and it must be $PTC_i' = PTC_i \oplus RPW_i \oplus h(r||PW_i')$. However, when the smart card replaces PTC_i' with PTC_i in its memory, this results wrong update of the new password PW_i' in the smart card.
As pointed out in [10], when the same user U_i will login later in the system providing his/her new changed password PW_i^*, we see that the login request of the same user U_i will be always rejected by the GWN even if U_i enters correctly PW_i^* in that time. Hence, due to such serious flaw in the design of Jiang et al.'s scheme, this effect will continue in all subsequent password change phases. As a result, this irrecoverable error enforces the user U_i to issue another new smart card by the GWN and register with the GWN again.

4.5 Lack of supporting dynamic node addition

A most desirable property of user authentication schemes designed for WSNs is the dynamic node addition after initial deployment of sensor nodes in a target field [12]. Wireless sensor networks often operate in an unattended environment. An adversary may physically capture some sensors to compromise their stored sensitive secret data and codes from their memory as they are not generally equipped with tamper-resistant hardware. Moreover, some sensor nodes may expire due to energy problem, because sensors are equipped with battery. Thus, we often require to deploy some new nodes in the network. It is noted that Jiang et al.'s scheme does not support this important feature.

4.6 Lack of formal security verification

Jiang et al.'s scheme contains only informal security analysis and it lacks a rigorous formal security proof for its security. Further, it lacks the formal security verification using some widely-accepted AVISPA tool [2].

5 The proposed scheme

In this section, we describe an improvement of Jiang et al.'s scheme in order to withstand the drawbacks found in their scheme. Our proposed scheme is a three-factor authentication scheme, which uses a user's personal biometrics as compared to Jiang et al.'s two-factor authentication scheme. The three-factor used in our scheme are (i) the smart card of a user, (ii) user's password, and (iii) user's personal biometrics. Traditional remote user authentication scheme based on the passwords may be vulnerable to the offline password guessing attacks, when a user picks up the chosen password from a small dictionary for the easy-to-remember purpose. In recent years, the research demonstrates that biometric-based remote user authentication schemes using passwords and smart cards are more secure and reliable, and hence this active research area has drawn a considerable research attention [8, 10, 23, 25, 26, 42]. Li and Hwang listed the following advantages of using biometric keys (e.g., fingerprints, faces, palm-prints, irises, and hand geometry) [25]:

- Biometric keys can not be lost or forgotten.
- Biometric keys are very difficult to copy or share.
- Biometric keys are extremely hard to forge or distribute.
- Biometric keys can not be guessed easily.
- Someone's biometrics is not easy to break than others.

In this paper, we use the user's personal biometric in our scheme to strengthen the security and other features which are not provided by Jiang et al.'s scheme. Our scheme keeps the original merits of Jiang et al.'s scheme. The various phases of our proposed scheme are described in the following subsections.

5.1 Pre-deployment of sensor nodes phase

This phase is executed by the GWN in offline prior to deployment of the sensor nodes in the target field. It consists of the following steps:

Step PD1. For each deployed sensor node SN_j, the GWN selects a unique identifier ID_{SN_j}.
Step PD2. The GWN generates randomly a large 1024-bit number K_{GWN-S}, which is considered as the GWN's private key only known to the GWN. After that for each

deployed sensor node SN_j, the GWN computes $TC_j = h(K_{GWN-S}\ ||ID_{SN_j})$, which is the temporal credential for SN_j.

Step PD3. Finally, each deployed sensor node SN_j is pre-loaded with the information TC_j as its temporal credential prior to its deployment in the target field.

Thus, it is clear that the pre-deployment of sensor nodes in our scheme is performed in offline. However, in Jiang et al.'s scheme this task is performed after deployment of sensor nodes. As a result, our scheme is more efficient as compared to Jiang et al.'s scheme, and our scheme thus helps to avoid a lot of communication and computational overheads by the sensor nodes after their deployment in the target field.

5.2 Registration phase

To register to the GWN in WSNs by a legal user U_i, the following steps need to be executed:

Step R1. U_i first selects a unique identity ID_i and chooses a password PW_i.

Step R2. U_i generates randomly a large 1024-bit secret number K. U_i computes the masked password $RPW_i = h(ID_i||K\ ||PW_i)$ and sends the registration request message $\langle ID_i, RPW_i\rangle$ to the GWN via a secure channel.

Step R3. After receiving the message in Step R2, the GWN continues to generate randomly a large 1024-bit private key K_{GWN-U}, and computes the temporal credential for the user U_i as $TC_i = h(K_{GWN-U}||ID_i||TE_i)$ and $PTC_i = TC_i \oplus RPW_i$, where TE_i is the expiration time of U_i's temporal credential TC_i. After that the GWN further randomly generates a large 1024-bit secret information X_s and computes $r_i = h(ID_i||X_s)$. The GWN selects a temporary identity TID_i for the user U_i and initializes it and then stores the tuple (TID_i, ID_i, TE_i) in the verification table. Finally, the GWN issues a smart card SC_i for the user U_i, which contains the information $\{h(\cdot), TID_i, TE_i, PTC_i, r_i\}$ and sends it to U_i via a secure channel.

Step R4. U_i imprints the biometric information B_i at the terminal of a specific device. U_i applies the fuzzy extractor function $Gen(\cdot)$ to generate secret key data σ_i and a public reproduction parameter τ_i as $Gen(B_i) = (\sigma_i, \tau_i)$ as in [19]. U_i then computes

$$\begin{aligned}
e_i &= h(ID_i||\sigma_i) \oplus K,\\
f_i &= h(ID_i||RPW_i||\sigma_i),\\
r_i^* &= r_i \oplus h(ID_i||K)\\
&= h(ID_i||X_s) \oplus h(ID_i||K).
\end{aligned}$$

U_i stores e_i, f_i, the fuzzy extractor functions $Gen(\cdot)$ and $Rep(\cdot)$, t and τ_i in the smart card. Further, U_i replaces r_i with r_i^* in the smart card. Finally, the smart card SC_i of U_i contains the information $\{h(\cdot), TID_i, TE_i, PTC_i, r_i^*, f_i, e_i, Gen(\cdot), Rep(\cdot), t, \tau_i\}$. Note that in our registration phase, the user U_i's secret number K is not stored in the smart card SC_i.

The registration phase of our scheme is summarized in Table 2.

5.3 Login phase

In order to login to the GWN, the user U_i needs to execute the following steps:

Step L1. U_i first inserts his/her smart card SC_i into a card reader and then enters his/her identity ID_i, password PW_i, and imprints the biometric information B_i at the sensor of the card reader. As described in [36] if the user U_i plans to use a mobile device to login the GWN, U_i can use the scan software of the mobile device to obtain B_i, and input $\{ID_i, PW_i, B_i\}$ into the login interface of the system.

Step L2. SC_i computes $\sigma_i^* = Rep(B_i, \tau_i)$ using B_i, and the parameters t and τ_i stored in its memory. SC_i further

Table 2 Registration phase of our scheme

User (U_i)/Smart Card (SC_i)	GWN
Inputs ID_i, PW_i, B_i. Generates a random secret number K. Computes $RPW_i = h(ID_i\|\|K\ \|\|PW_i)$. $\xrightarrow{\langle ID_i, RPW_i\rangle}$ (via a secure channel)	
	Generates private key K_{GWN-U}. Computes $TC_i = h(K_{GWN-U}\|\|ID_i\|\|TE_i)$, $PTC_i = TC_i \oplus RPW_i$. Generates secret information X_s and computes $r_i = h(ID_i\|\|X_s)$. Selects temporary identity TID_i of U_i and initializes it. Stores the tuple (TID_i, ID_i, TE_i) in verification table. $\xleftarrow{\langle Smart\,Card(h(\cdot), TID_i, TE_i, PTC_i, r_i)\rangle}$ (via a secure channel)
Computes $Gen(B_i) = (\sigma_i, \tau_i)$, $e_i = h(ID_i\|\|\sigma_i) \oplus K$, $f_i = h(ID_i\|\|RPW_i\|\|\sigma_i)$, $r_i^* = r_i \oplus h(ID_i\|\|K)$. Replaces r_i with r_i^* in smart card. Stores $e_i, f_i, Gen(\cdot), Rep(\cdot)$, t and τ_i in smart card.	

computes $K^* = e_i \oplus h(ID_i||\sigma_i^*)$, $RPW_i^* = h(ID_i||K^*||PW_i)$ and $f_i^* = h(ID_i||RPW_i^*||\sigma_i^*)$, and then checks if $f_i^* = f_i$ holds or not. If it does not hold, this ensures that U_i enters his/her password incorrectly and does not pass both biometric and password verifications. Otherwise, SC_i proceeds to the next step.

Step L3. SC_i generates a current timestamp TS_1 and randomly chooses a temporary key K_i. SC_i then computes $TC_i = PTC_i \oplus RPW_i^*$, $M_1 = r_i^* \oplus h(ID_i||K^*) = h(ID_i||X_s)$, $PKS_i = K_i \oplus h(TC_i||M_1||TS_1)$ and $C_i = h(ID_i||K_i||TC_i||M_1||TID_i||TS_1)$. Finally, SC_i sends the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ to the GWN via a public channel.

The login phase of our scheme is summarized in Table 3.

5.4 Authentication and key agreement phase

After receiving the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ by the GWN from U_i, the following steps are executed for mutual authentication and key establishment:

Step AK1. The GWN checks the timeliness of TS_1 in the received message by the condition $|T_{GWN}^* - TS_1| < \Delta T$, where T_{GWN}^* is the current timestamp of the GWN. If it is not valid, the GWN terminates this current session and sends the reject message REJ back to U_i. Otherwise, the GWN obtains the real identity ID_i of U_i corresponding to TID_i in its verification table. The GWN then computes $M_2 = h(ID_i||X_s)$ using its own secret information X_s and retrieved ID_i, $TC_i = h(K_{GWN-U}||ID_i||TE_i)$, $K_i = PKS_i \oplus h(TC_i||M_2||TS_1)$, $C_i^* = h(ID_i||K_i||TC_i||M_2||TID_i||TS_1)$, and then checks the condition $C_i^* = C_i$. If this condition fails, the GWN terminates this session immediately and sends REJ message back to U_i. Otherwise, the GWN confirms that U_i is a legitimate user, and computes the accessed sensor node SN_j's temporal credential $TC_j = h(K_{GWN-S}||ID_{SN_j})$, $C_{GWN} = h(TID_i||TC_j||TS_2)$, and $PKS_{GWN} = (K_i \oplus M_2) \oplus h(TC_j||TS_2)$, where TS_2 is the current timestamp of the GWN. Finally, the GWN sends the authentication request message $\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle$ to SN_j via a public channel.

Step AK2. When the sensor node SN_j receives the message $\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle$ from the GWN, SN_j first checks the validity of TS_2 by the condition $|T_j^* - TS_2| < \Delta T$, where T_j^* is the current time of SN_j. If it is not valid, SN_j terminates this phase immediately. Otherwise, SN_j continues to compute $C_{GWN}^* = h(TID_i\ ||TC_j||TS_2)$ using its stored TC_j and received TID_i and TS_2. If the validation of $C_{GWN}^* = C_{GWN}$ does not hold, SN_j rejects this message. On the other hand, SN_j confirms that the GWN is legitimate. SN_j further continues to compute $M_3 = PKS_{GWN} \oplus h(TC_j\ ||TS_2) = K_i \oplus M_2 = K_i \oplus h(ID_i||X_s)$. After that SN_j generates a random temporary key K_j and computes $C_j = h(K_j||TID_i||ID_{SN_j}||TS_3)$ and $PKS_j = K_j \oplus h(M_3\ ||TS_3)$, where TS_3 is the current timestamp of SN_j. SN_j then sends the the authentication reply message $\langle ID_{SN_j}, TS_3, C_j, PKS_j\rangle$ to the GWN via a public channel.

Step AK3. After receiving the message in Step AK2, the GWN computes $K_j = PKS_j \oplus h((K_i \oplus M_2)||TS_3)$ using the received TS_3 and its previously computed K_i and M_2 in step AK1. The GWN then computes $C_j^* = h(K_j||TID_i||ID_{SN_j}||TS_3)$ and checks if $C_j^* = C_j$. If it does not hold, the GWN rejects this message. Otherwise, the GWN confirms that SN_j is a legitimate sensor node in WSN. The GWN then generates a new temporary identity TID_i^{new} and computes $D_{GWN} = TID_i^{new} \oplus h((K_i \oplus M_2)||TS_3||TS_4)$, where TS_4 is the current time of the GWN. The GWN replaces TID_i with TID_i^{new} in its verification table. The GWN also computes $E_{GWN} = h(ID_i||ID_{SN_j}||TC_i||D_{GWN}||K_j||TS_3||TS_4)$. The GWN then sends the acknowledgment message $\langle ID_{SN_j}, TS_3, TS_4, PKS_j, D_{GWN}, E_{GWN}\rangle$ to the user U_i via a public channel.

Table 3 Login phase of our scheme

User (U_i)/Smart Card (SC_i)	GWN
Inserts smart card and inputs ID_i, PW_i, B_i. Computes $\sigma_i^* = Rep(B_i, \tau_i)$, $K^* = e_i \oplus h(ID_i\|\|\sigma_i^*)$, $RPW_i^* = h(ID_i\|\|K^*\ \|\|PW_i)$ and $f_i^* = h(ID_i\|\|RPW_i^*\|\|\sigma_i^*)$. Checks if $f_i^* = f_i$? If so, generates a current timestamp TS_1, temporary key K_i, and computes $TC_i = PTC_i \oplus RPW_i^*$, $M_1 = r_i^* \oplus h(ID_i\|\|K^*) = h(ID_i\|\|X_s)$, $PKS_i = K_i \oplus h(TC_i\|\|M_1\|\|TS_1)$, $C_i = h(ID_i\|\|K_i\|\|TC_i\|\|M_1\|\|TID_i\|\|TS_1)$. $\xrightarrow{\langle TID_i, C_i, PKS_i, TS_1\rangle}$ (via a public channel)	

Step AK4. After receiving the message in Step AK3, the smart card SC_i of the user U_i checks the timeliness of TS_4. If it is valid, SC_i computes $TID_i^{new} = D_{GWN} \oplus h((K_i \oplus M_1)||TS_3||TS_4)$, $K_j = PKS_j \oplus h((K_i \oplus M_1)||TS_3)$, and $E^*_{GWN} = h(ID_i||ID_{SN_j}||TC_i||D_{GWN} ||K_j||TS_3||TS_4)$. If the condition $E^*_{GWN} = E_{GWN}$ passes, SC_i (that is, the user U_i) confirms that both SN_j and GWN are legitimate nodes in WSN. After mutual authentication between U_i and SN_j, SC_i of U_i computes the shared session key $SK_{ij} = h((K_i \oplus M_1) \oplus K_j)$ with the accessed sensor node SN_j. Similarly, SN_j computes the session key shared with U_i as $SK^*_{ij} = h(M_3 \oplus K_j)$, which is equal to $h((K_i \oplus K_2) \oplus K_j) = h((K_i \oplus M_1) \oplus K_j = SK_{ij}$, since $M_1 = M_2$. Thus, it is clear that both U_i and SN_j have the same session key, that is, $SK_{ij} = SK^*_{ij}$.
Finally, U_i and SN_j can use SK_{ij} for their future secure communications. Note that as compared to Jiang et al.'s scheme, we have used the secret credential X_s of the GWN in the computation of SK_{ij} and SK^*_{ij}, because $M_1 = M_2 = h(ID_i||X_s)$.

The authentication and key agreement phase of our scheme is summarized in Table 4.

5.5 Password and biometric update phase

For security reasons, it is desirable that a legal user U_i should be allowed to change/update his/her password as well as personal biometrics. This procedure should be done by the smart card of the user U_i without involving the GWN, which needs to be performed locally so that the GWN must

Table 4 Authentication and key agreement phase of our scheme

User (U_i)/Smart Card (SC_i)	GWN	Sensor node (SN_j)
	Checks the timeliness of TS_1. If valid, computes $M_2 = h(ID_i\|\|X_s)$, $TC_i = h(K_{GWN-U}\|\|ID_i\|\|TE_i)$, $K_i = PKS_i \oplus h(TC_i\|\|M_2\|\|TS_1)$. $C_i^* = h(ID_i\|\|K_i\|\|TC_i\|\|M_2\|\|TS_1)$. Checks if $C_i^* = C_i$. If so, computes $TC_j = h(K_{GWN-S}\|\|ID_{SN_j})$, $C_{GWN} = h(TID_i\|\|TC_j\|\|TS_2)$, $PKS_{GWN} = (K_i \oplus M_2)$ $\oplus h(TC_j\|\|TS_2)$. $\xrightarrow{\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle}$ (via a public channel)	
		Checks if $\|T_j^* - TS_2\| < \Delta T$? If it is valid, computes $C^*_{GWN} = h(TID_i\ \|\|TC_j\|\|TS_2)$. Checks if $C^*_{GWN} = C_{GWN}$? If it holds, computes $M_3 = PKS_{GWN} \oplus h(TC_j\|\|TS_2)$, $C_j = h(K_j\|\|TID_i\|\|ID_{SN_j}\|\|TS_3)$ $PKS_j = K_j \oplus h(M_3\ \|\|TS_3)$. $\xleftarrow{\langle ID_{SN_j}, TS_3, C_j, PKS_j\rangle}$ (via a public channel)
	Computes $K_j = PKS_j \oplus h((K_i \oplus M_2)\|\|TS_3)$, $C_j^* = h(K_j\|\|TID_i\|\|ID_{SN_j}\|\|TS_3)$. Verifies if $C_j^* = C_j$? If it is valid, generates TID_i^{new} and computes $D_{GWN} = TID_i^{new} \oplus h((K_i \oplus M_2)\|\|TS_3\|\|TS_4)$. Updates TID_i with TID_i^{new}, and computes $E_{GWN} = h(ID_i\|\|ID_{SN_j}\|\|TC_i\|\|D_{GWN}$ $\|\|K_j\|\|TS_3\|\|TS_4)$. $\xleftarrow{\langle ID_{SN_j}, TS_3, TS_4, PKS_j, D_{GWN}, E_{GWN}\rangle}$ (via a public channel)	
Checks the timeliness of TS_4. If it is valid, computes $TID_i^{new} = D_{GWN} \oplus h((K_i \oplus M_1)$ $\|\|TS_3\|\|TS_4)$, $K_j = PKS_j \oplus h((K_i$ $\oplus M_1)\|\|TS_3)$, $E^*_{GWN} = h(ID_i\|\|$ $ID_{SN_j}\|\|TC_i\|\|D_{GWN}\ \|\|K_j\|\|TS_3$ $\|\|TS_4)$. Checks if $E^*_{GWN} = E_{GWN}$? If it passes, computes the session key $SK_{ij} = h((K_i \oplus M_1) \oplus K_j)$.		Computes the session key $SK^*_{ij} = h(M_3 \oplus K_j)$ $= h((K_i \oplus M_1) \oplus K_j)$.

not be aware that the user U_i has changed his/her password and biometrics.

Suppose U_i wants to update his/her old password PW_i^{old} and old biometrics B_i^{old}. The following steps are then executed:

Step PBC1. U_i first inserts his/her smart card SC_i into the card reader of a terminal. U_i then inputs ID_i, PW_i^{old} and also imprints B_i^{old} at the terminal.

Step PBC2. SC_i then computes $\sigma_i^{old} = Rep(B_i^{old}, \tau_i)$, $K^* = e_i \oplus h(ID_i||\sigma_i^{old})$, $RPW_i^{old} = h(ID_i||K^*|| PW_i^{old})$, and $f_i^{old} = h(ID_i||RPW_i^{old}||\sigma_i^{old})$. SC_i then checks if $f_i^{old} = f_i$. If it holds, it ensures that U_i has entered his/her correct old password PW_i^{old} and old biometrics B_i^{old}. Otherwise, this phase is terminated immediately.

Step PBC3. SC_i then asks U_i to enter his/her new password PW_i^{new} and imprint new biometrics B_i^{new} at the terminal. SC_i computes $x = PTC_i \oplus RPW_i^{old} = TC_i \oplus RPW_i \oplus RPW_i^{old} = TC_i$, $RPW_i^{new} = h(ID_i||K^* ||PW_i^{new})$, $PTC_i^{new} = x \oplus RPW_i^{new}$, which is equal to $TC_i \oplus RPW_i^{new}$. SC_i further computes $Gen(B_i^{new}) = (\sigma_i^{new}, \tau_i^{new})$, $e_i^{new} = h(ID_i||\sigma_i^{new}) \oplus K^*$, $f_i^{new} = h(ID_i||RPW_i^{new}||\sigma_i^{new})$.

Step PBC4. Finally, SC_i replaces PTC_i, f_i, e_i, and τ_i with PTC_i^{new}, f_i^{new}, e_i^{new}, and τ_i^{new}, respectively.

Thus, it is clear from our password and biometric update phase that the smart card SC_i updates the new password and biometrics only when the user passes both old password and biometric verifications. Further, this phase is also executed locally without the knowledge of the GWN.

5.6 Dynamic node addition phase

Suppose a new sensor node SN_j^{new} is to be deployed in the existing sensor network. For this purpose, the following steps are executed by the GWN in offline prior to its deployment in the target field:

Step DA1. The GWN first assigns a unique random identity $ID_{SN_j}^{new}$ for SN_j^{new}.

Step DA2. The GWN then computes the temporal credential for SN_j^{new} as $TC_i^{new} = h(K_{GWN-S}||ID_{SN_j}^{new})$.

Step DA3. Finally, the GWN loads TC_i^{new} in the memory of SN_j^{new} prior to its deployment.

After deployment of the new sensor node SN_j^{new} in the target field, the GWN needs to inform the user U_i so that he/she can access the real-time data from it later.

6 Security analysis of the proposed scheme

In this section, we analyze the security of our proposed scheme. Through both the rigorous informal and formal security analysis, we show that our scheme has the ability to tolerate various known attacks, which are outlined in the following subsections.

6.1 Informal security analysis

We show that our scheme resists the following attacks.

6.1.1 Privileged insider attack

In our proposed scheme, during the registration phase the user U_i sends his/her identity ID_i and pseudo-password $RPW_i = h(ID_i||K||PW_i)$ to the GWN via a secure channel. As in Section 4.1, we assume that the insider of the GWN has the lost/stolen smart card SC_i of U_i. Then, using the power analysis attack [24, 27], the insider can extract the information $TID_i, TE_i, PTC_i, r_i^*, f_i, e_i, t, \tau_i$ from SC_i, where $PTC_i = TC_i \oplus RPW_i$, $e_i = h(ID_i||\sigma_i) \oplus K$, $r_i^* = h(ID_i||X_s) \oplus h(ID_i||K)$, and $f_i = h(ID_i||RPW_i||\sigma_i)$. The insider of the GWN can compute $u = PTC_i \oplus TC_i$ and $v = r_i^* \oplus h(ID_i||X_s) = h(ID_i|| K)$. Since K is an 1024-bit large secret number only known to U_i, due to collision-resistant property of a one-way hash function $h(\cdot)$ it is a computationally infeasible task for the insider attacker to derive K from v, and as a result, to correctly guess PW_i without knowing K though the attacker knows ID_i and RPW_i. Hence, the attacker has no way to derive PW_i. Thus, our scheme protects the privileged-insider attack.

6.1.2 Online password and biometric key guessing attack

Suppose an adversary intercepts all the messages $\langle TID_i, C_i, PKS_i, TS_1\rangle$, $\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle$, $\langle ID_{SN_j}, TS_3, C_j, PKS_j\rangle$, and $\langle ID_{SN_j}, TS_3, TS_4, PKS_j, D_{GWN}, E_{GWN}\rangle$ during the login phase and the authentication and key agreement phase of our scheme. It is clear that none of the messages involves the password and biometric key data of the user U_i. The adversary then can not derive PW_i and σ_i from these messages. Hence, our scheme protects online password and biometric key guessing-attack.

6.1.3 Offline password and biometric key guessing attack

Assume that an adversary has the stolen smart card SC_i of a legal user U_i. According to our threat model in Section 1.2, the adversary can extract the sensitive information $TID_i, TE_i, PTC_i, r_i^*, f_i, e_i, t, \tau_i$ from SC_i.

Without knowing the biometric B_i of the user U_i, the adversary cannot derive σ_i using τ_i. Furthermore, the adversary cannot derive K from e_i without knowing both σ_i and ID_i. Hence, it is a computationally infeasible task for the adversary to guess correctly PW_i from f_i without knowing K, ID_i and σ_i due to the collision-resistant property of the one-way hash function $h(\cdot)$. In this way, our scheme is resilient against deriving the password and biometric key data σ_i of U_i. Thus, the offline password and biometric key guessing attack is eliminated from our scheme.

6.1.4 Replay attack

Suppose an adversary intercepts the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ during the login phase, and replies the same message to the GWN after some time. When the GWN receives this message, it checks the timeliness of TS_1 in the received message by the condition $|T^*_{GWN} - TS_1| < \Delta T$, where T^*_{GWN} is the current timestamp of the GWN. If it is not valid, the GWN ensures that the received message is not a fresh message and it will immediately discards this message. Due to the timestamps used in the other messages between U_i, GWN and SN_j, those will be also detected as duplicate messages, if the adversary sends those same messages later. The replay attack is then prevented in our design.

6.1.5 Man-in-the-middle attack

Suppose an adversary intercepts the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ during the login phase, and tries to modify this message in order to convince the GWN that the modified message is a valid login request message. Note that the adversary has TID_i and TS_1 from the login request message. However, the adversary does not know ID_i, K_{GWN-U}, TE_i and X_s. In order to compute $M_1 = h(ID_i\,||X_s)$ and $TC_i = h(K_{GWN-U}||ID_i||TE_i)$, the adversary needs to know ID_i, X_s, K_{GWN-U} and TE_i. Though the adversary can generates a random temporary key K'_i and use a different timestamp TS'_1, he/she is still unable to compute $PKS'_i = K'_i \oplus h(TC_i||M_1||TS'_1)$ and $C'_i = h(ID_i||K'_i\,||TC_i||M_1||TID_i||TS'_1)$ because these computations involve M_1 and TC_i, the temporal credential of the user U_i. As a result, the adversary does not have any ability to change the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ to another modified message $\langle TID_i, C'_i, PKS'_i, TS'_1\rangle$. Similarly, the adversary cannot be also successful to modify other messages during the authentication and key agreement phase. Hence, our scheme tolerates the man-in-the-middle attack against the adversary.

6.1.6 Stolen-verifier attack

In this attack, an attacker tries to steal or modify the verifiers (e.g., plaintext passwords, hashed passwords, biometric key data, hashed biometric key data) stored in the GWN. Since our scheme does not maintain any authentication information derived from U_i's password PW_i, the attacker has no way to steal or modify any authentication information from the GWN. Thus, our scheme is resilient against stolen-verifier attack.

6.1.7 Forgery attacks

We consider the following three cases:

U_i **forgery attack:** Suppose an adversary eavesdrop or intercept the login request message of the previous sessions. Then the adversary can forge a forged message and send it to the GWN. After receiving the forged message, the GWN can extract K_i with the secret $TC_i = h(K_{GWN-U}||ID_i||TE_i)$, and then compute $M_2 = h(ID_i||X_s)$ and $C^*_i = h(ID_i||K_i\,||TC_i||M_1||TID_i||TS_1)$. The GWN can verify the legitimacy of the user U_i by verifying the condition $C^*_i = C_i$. However, as stated in Section 6.1.5, the adversary will have any ability to forge the login request message without knowing ID_i, X_s, K_{GWN-U} and TE_i. For this purpose, the adversary needs to compute a valid C_i. Thus, the adversary cannot be passed by the GWN and hence, our scheme is secure against this attack.

GWN forgery attack: It is clear from our authentication and key agreement phase that the authentication request message $\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle$ is protected by the collision-resistant one-way hash function $h(\cdot)$. Without knowing $TC_j = h(K_{GWN-S}||ID_{SN_j})$ the adversary cannot compute C_{GWN} and PKS_{GWN}. Furthermore, without knowing $TC_i = h(K_{GWN-U}||ID_i||TE_i)$ the adversary cannot compute E_{GWN}. The adversary does not have any ability forge the messages $\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle$ and $\langle ID_{SN_j}, TS_3, TS_4, PKS_j, D_{GWN}, E_{GWN}\rangle$. Our scheme is then secure against GWN forgery attack.

SN_j **forgery attack:** To forge the message $\langle ID_{SN_j}, TS_3, C_j, PKS_j\rangle$ in Step AK2, the adversary needs to know $TC_j = h(K_{GWN-S}||ID_{SN_j})$ in order to compute PKS_j. Thus, the adversary does not have any ability forge this message and our scheme is thus secure against SN_j forgery attack.

6.1.8 Many logged-in users with the same login-id attack

Assume that two users U_i and U_j pick up the same password PW. Then the hash values $f_i = h(ID_i||RPW_i||\sigma_i)$ and $f_j = h(ID_j||RPW_j||\sigma_j)$ are distinct due to the properties of personal biometrics, random numbers selected by the users U_i and U_j, respectively, and ID_i and ID_j. To login to the GWN, a user U_i must have a valid $\langle ID_i, PW_i, B_i\rangle$ and a smart card corresponding to these information. Since our scheme requires on-card computation to login to the GWN, once the smart card is removed from the system, the login session immediately terminates. As a result, our scheme prevents the many logged-in users with the same login-id attack.

6.1.9 Identity guessing attack

From our scheme, it is noted that the real identity ID_i of a legal user U_i is only known to that user U_i and the GWN, which stores ID_i in its verification table. However, ID_i is never transmitted over the public channel for authentication purpose. Instead of that, the random temporary identity TID_i is used in the public communications. Thus, the adversary does not have any ability to derive ID_i for verifying his/her guessing identity. As a result, our scheme is resilient against the identity guessing attack.

6.1.10 Tracing attack

In our design, as in Jiang et al.'s scheme, the temporary identity TID_i is purely generated randomly and it is also updated after each successful authentication session. Since each temporary identity for each session is distinct and random, the adversary has no way to trace a specific user during the communications. Our scheme is then resilient against the tracing attack.

6.1.11 Password and biometric change attack

In order to change the password and biometric of a legal user U_i, an adversary needs to pass both biometric and password verifications. Without those, the attacker does not have any ability to update the new password and biometric in the smart card. Our scheme is thus secure against the password and biometric change attack.

6.1.12 User anonymity and unlinkability

As in Jiang et al.'s scheme, in our scheme a legal user U_i's real identity ID_i is never transmitted over an insecure channel. It is difficult task for an adversary to know the real identity of U_i from the protocol messages during the login phase and the authentication and key agreement phase. Further, a different temporary identity TID_i is used for each session in order to keep the privacy of U_i. Since TID_i is different and random in each session, it is then unlinkable. Thus, any adversary and sensor nodes do not have any idea that who is communicating with the GWN. As a result, it is difficult for the adversary to detect whether two protocols runs involve the same user U_i. Hence, our scheme prevents the leakage of user identity and protects user's privacy.

6.1.13 Three-factor security

In the three-factor security model, the main goals of an attacker are to mount an impersonation attack where the attacker has learned at most two elements of the triple $\{PW_i, SC_i, B_i\}$, in order to obtain the last element or to compromise the user anonymity. As in the analysis of Tan's scheme, it is also clear that our scheme provides the three-factor security.

6.1.14 Resilience against node capture attack

As described in [12], we measure the resilience against node capture attack of a user authentication scheme in WSN by estimating the fraction of total secure communications that are compromised by a capture of c sensor nodes *not including* the communication in which the compromised nodes are directly involved. In other words, we want to find out the effect of c sensor nodes being compromised on the rest of the network. For example, for any non-compromised sensor nodes SN_j, we need to find out the probability that the adversary can decrypt the secure communication between SN_j and a legal user U_i when c sensor nodes are already compromised. Let $P_e(c)$ denote the probability that the adversary compromises a fraction of total secure communications using c compromised nodes. If $P_e(c) = 0$, our scheme is called unconditionally secure against node capture attack. Assume that the adversary captures a sensor node SN_j. Then the attacker will know the secret temporal credential TC_j stored in its memory because the sensor nodes or cluster heads are not equipped with tamper-resistant hardware, and the session key SK_{ij} shared between that sensor node SN_j and the user U_i is also revealed to that adversary. Note that each sensor node is pre-loaded with a unique key TC_j prior to its deployment, because $TC_j = h(K_{GWN-S}||ID_{SN_j})$ and each ID_{SN_j} is distinct. Moreover, it is computationally infeasible to derive the 1024-bit private key K_{GWN-S} due to the collision-resistant property of the one-way hash function $h(\cdot)$. Further, each sensor node establishes a distinct secret session key with a user, which is different from all other session keys in the network. As a result, the adversary has only the ability to respond with false data to a legitimate user by capturing a sensor node from which the user wants to access data. However, other non-captured sensor nodes have the ability to communicate with 100 % secrecy with the actual real-time data to the legitimate users. The compromise of a sensor node does not lead to compromise of any other secure communication between the user and the non-compromised sensor nodes in the network. Thus, our scheme is unconditionally security against node capture attack.

6.2 Formal security analysis

In this section, we show through the formal security analysis that our scheme is secure against an adversary. We follow the formal security analysis of our scheme similar to that in [10, 11, 29]. We have used the method of contradiction proof [7] for our formal security analysis. Note that one can also prove the formal security in the standard model.

For this purpose, we assume that there exists the following oracle for the attacker (adversary), say $\mathcal{A}$:

– *Reveal* : This is an oracle, which unconditionally outputs the input string x from the corresponding hash value $y = h(x)$.

Theorem 1 *Under the assumption that a one-way hash function $h(\cdot)$ closely behaves like an oracle, our scheme is secure against an adversary for deriving the identity ID_i, the password PW_i, the biometric key data σ_i of a legal user U_i, and the secret information X_s of the GWN, even if the user U_i's smart card SC_i is lost/stolen.*

Proof In this proof, we need to construct an adversary, say $\mathcal{A}$, who will have the ability to derive the identity ID_i, the password PW_i, the biometric key data σ_i of a legal user U_i, and the secret information X_s of the GWN. We assume that the adversary $\mathcal{A}$ has the lost/stolen smart card SC_i and can extract all the sensitive information from its memory using the power analysis attack [24, 27]. For this purpose, we assume that $\mathcal{A}$ has access to the oracle *Reveal*, which gives the input string x from the corresponding hash value $y = h(x)$. $\mathcal{A}$ then uses the *Reveal* oracle for running an experimental algorithm, say $EXP1^{HASH}_{UAS,\mathcal{A}}$ for our proposed user authentication scheme, say UAS, which is provided in Algorithm 1. The success probability for $EXP1^{HASH}_{UAS,\mathcal{A}}$ can be defined as $Succ1^{HASH}_{UAS,\mathcal{A}} = |Pr[EXP1^{HASH}_{UAS,\mathcal{A}} = 1] - 1|$, where $Pr[E]$ is the probability of an event E. The advantage function for this experiment then becomes $Adv1(et_1, q_R) = \max_{\mathcal{A}}\{Succ1^{HASH}_{UAS,\mathcal{A}}\}$, where the maximum is taken over all $\mathcal{A}$ with execution time et_1, and the number of queries q_R made to the *Reveal* oracle. We call our scheme is secure against an adversary $\mathcal{A}$ for deriving ID_i, PW_i, σ_i and X_s, if $Adv1\ (et_1, q_R) \leq \epsilon$, for any sufficiently small $\epsilon > 0$. According to this experiment given in Algorithm 1, it is clear that if $\mathcal{A}$ has the ability to invert the one-way hash function $h(\cdot)$, then only he/she can easily derive ID_i, PW_i, σ_i and X_s, and win the game. However, it is a computationally infeasible problem to invert $h(\cdot)$, that is, $Adv^{HASH}_{\mathcal{A}}(t_1) \leq \epsilon_1$, for any sufficiently small $\epsilon_1 > 0$ (provided in Definition 1). Hence, we have $Adv1\ (et_1, q_R) \leq \epsilon$, since $Adv1\ (et_1, q_R)$ depends on the advantage $Adv^{HASH}_{\mathcal{A}}(t_1)$. Thus, our scheme is secure against an adversary for deriving ID_i, PW_i, σ_i and X_s, even if the user U_i's smart card SC_i is lost/stolen.

Algorithm 1 $EXP1^{HASH}_{UAS,\mathcal{A}}$

```
1: Extract the information $TID_i, TE_i, PTC_i, r_i^*, f_i$, and $e_i$ from the memory of the smart card $SC_i$ using the power analysis attack [24], [27], where $f_i = h(ID_i||RPW_i||\sigma_i)$, $r_i^* = r_i \oplus h(ID_i||K)$, $e_i = h(ID_i||\sigma_i) \oplus K$, $RPW_i = h(ID_i||K||PW_i)$.
2: Call $Reveal$ oracle on input $f_i$ to retrieve the information $ID_i$, $RPW_i$ and $\sigma_i$ as $(ID_i'||RPW_i'||\sigma_i') \leftarrow Reveal(f_i)$.
3: Compute $K' = e_i \oplus h(ID_i'||\sigma_i')$.
4: Compute $u = r_i^* \oplus h(ID_i'||K')$.
5: Intercept the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$, where $C_i = h(ID_i||K_i||TC_i||M_1||TID_i||TS_1)$.
6: Call $Reveal$ oracle on input $C_i$ to retrieve the information $ID_i$, $K_i$, $TC_i$, $M_1$, $TID_i$ and $TS_1$ as $(ID_i^*||K_i^*||TC_i^*||M_1^*||TID_i^*||TS_1^*) \leftarrow Reveal(C_i)$.
7: if $(ID_i* = ID_i')$ and $(TID_i^* = TID_i)$ and $(TS_1^* = TS_1)$ and $(u = M_1^*)$ then
8:     Call $Reveal$ oracle on input $u$ to retrieve $ID_i$ and $X_s$ as $(ID_i^{**}||X_s^{**}) \leftarrow Reveal(u)$.
9:     Call $Reveal$ oracle on input $RPW_i'$ to retrieve the information $ID_i$, $K$ and $PW_i$ as $(ID_i^{***}||K^{***}||PW_i^{***}) \leftarrow Reveal(RPW_i')$.
10:    if $(ID_i^{**} = ID_i^{***})$ and $(K^{***} = K')$ then
11:        Accept $ID_i'$, $PW_i^{***}$ and $\sigma_i'$ as the correct identity $ID_i$, password $PW_i$, biometric key data $\sigma_i$ of the user $U_i$, and $X_s^{**}$ as the correct $X_s$ of the GWN.
12:        return 1 (Success)
13:    else
14:        return 0 (Failure)
15:    end if
16: else
17:    return 0 (Failure)
18: end if
```

□

Theorem 2 *Under the assumption that a one-way hash function $h(\cdot)$ closely behaves like an oracle, our scheme is secure against an adversary for deriving the session key SK_{ij} shared between a legal user U_i and an accessed sensor node SN_j in WSN.*

Proof The proof of this theorem is similar to that in Theorem 1. We need to construct an adversary, $\mathcal{A}$, who will have the ability to derive the session key SK_{ij} shared between a legal user U_i and an accessed sensor node SN_j in WSN. We also assume that there exits an oracle *Reveal* and $\mathcal{A}$

has access to that oracle $Reveal$, which gives the input string x from the corresponding hash value $y = h(x)$. $\mathcal{A}$ then uses the $Reveal$ oracle for running the experimental algorithm, say $EXP2^{HASH}_{UAS,\mathcal{A}}$ for our proposed user authentication scheme, UAS, which is provided in Algorithm 2. We define the success probability for $EXP2^{HASH}_{UAS,\mathcal{A}}$ as $Succ2^{HASH}_{UAS,\mathcal{A}} = |Pr[EXP2^{HASH}_{UAS,\mathcal{A}} = 1]-1|$. The advantage function for this experiment is then defined by $Adv2(et_2, q_R) = \max_{\mathcal{A}}\{Succ2^{HASH}_{UAS,\mathcal{A}}\}$, where the maximum is taken over all $\mathcal{A}$ with execution time et_2, and the number of queries q_R made to the $Reveal$ oracle. Our scheme is said to be secure against an adversary $\mathcal{A}$ for deriving SK_{ij}, if $Adv2\ (et_2, q_R) \leq \epsilon$, for any sufficiently small $\epsilon > 0$. Consider the experiment given in Algorithm 2. It is clear that if $\mathcal{A}$ has the ability to invert the one-way hash function $h(\cdot)$, then only he/she can easily derive SK_{ij}, and win the game. However, it is a computationally infeasible problem to invert $h(\cdot)$, because $Adv^{HASH}_{\mathcal{A}}(t_1) \leq \epsilon_1$, for any sufficiently small $\epsilon_1 > 0$ (provided in Definition 1). Hence, we have $Adv2\ (et_2, q_R) \leq \epsilon$, since $Adv2\ (et_2, q_R)$ is also dependent on the advantage $Adv^{HASH}_{\mathcal{A}}(t_1)$. As a result, our scheme is secure against an adversary for deriving SK_{ij}.

Algorithm 2 $EXP2^{HASH}_{UAS,\mathcal{A}}$

```
1: Intercept the login request message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ during the login phase.
2: Call $Reveal$ oracle on input $C_i$ to retrieve $ID_i$, $K_i$, $TC_i$, $M_1$, $TID_i$ and $TS_1$ as $(ID_i'||K_i'||TC_i'||M_1'||TID_i'||TS_1') \leftarrow Reveal(C_i)$.
3: if $TID_i' = TID_i$ and $(TS_1' = TS_1)$ then
4:   Compute $x = PKS_i \oplus h(TC_i'||M_1'||TS_1)$.
5:   if $(x = K_i')$ then
6:     Intercept the message $\langle ID_{SN_j}, TS_3, C_j, PKS_j\rangle$ during the authentication and key agreement phase.
7:     Call $Reveal$ oracle on $C_j$ to retrieve $K_j$, $TID_i$, $ID_{SN_j}$ and $TS_3$ as $(K_j''||TID_i''||ID_{SN_j}''||TS_3'') \leftarrow Reveal(C_j$.
8:     if $(TID_i'' = TID_i)$ and $(TS_3'' = TS_3)$ and $(ID_{SN_j}'' = ID_{SN_j})$ then
9:       Compute $SK_{ij}' = h((K_i' \oplus M_1') \oplus K_j'')$.
10:      Accept $SK_{ij}'$ as the correct session key shared between $U_i$ and $SN_j$.
11:      return 1 (Success)
12:    else
13:      return 0 (Failure)
14:    end if
15:  else
16:    return 0 (Failure)
17:  end if
18: else
19:   return 0 (Failure)
20: end if
```

□

7 Simulation results for formal security verification using AVISPA tool

In this section, we simulate our scheme for the formal security verification using the widely-accepted AVISPA (Automated Validation of Internet Security Protocols and Applications) tool and show that our scheme is secure against the replay and man-in-the-middle attacks against an adversary.

AVISPA is considered as a push-button tool for the automated validation of Internet security-sensitive protocols and applications. It provides a modular and expressive formal language for specifying protocols and their security properties, and integrates different back-ends that implement a variety of state-of-the-art automatic analysis techniques [1], [2]. AVISPA consists of the four backends, which are the OFMC (On-the-fly Model-Checker), CL-AtSe (Constraint-Logic based Attack Searcher), SATMC (SAT-based Model-Checker), and TA4SP (Tree Automata based on Automatic Approximations for the Analysis of Security Protocols). Protocols to be implemented by the AVISPA tool have to be specified in HLPSL (High Level Protocols Specification Language) [30], and written in a file with extension hlpsl. This language is based on roles: basic roles for representing each participant role, and composition roles for representing scenarios of basic roles. Each role is then independent from the other roles, which gets some initial information by parameters, and then communicate with the other roles by channels. The intruder is modeled using the Dolev-Yao model [16] (as described in our threat model in Section 1.2) with the possibility for the intruder can have a legitimate role in a protocol run. The role system also defines the number of sessions and the number of principals and the roles.

The output format (OF) is generated by using one of the four back-ends explained above. When the analysis of a protocol has been successful (by finding an attack or not), the output describes precisely what is the result, and under what conditions it has been obtained. In OF, we have the following sections:

– The first printed section, called the SUMMARY, indicates that whether the tested protocol is safe, unsafe, or whether the analysis is inconclusive.
– The second section, called DETAILS, either explains under what condition the tested protocol is declared safe, or what conditions have been used for finding an attack, or finally why the analysis was inconclusive.
– Other remaining sections PROTOCOL, GOAL and BACKEND are the name of the protocol, the goal

of the analysis and the name of the back-end used, respectively.
- Finally, after some comments and statistics, the trace of an attack (if any) is also printed in the standard Alice-Bob format.

Protocols need to be implemented in HLPSL. HLPSL supports various basic types, and some of them are given below for better understanding for the specifications of different roles provided in the next subsection:

- *agent:* It represents a principal name. The intruder is always assumed to have the special identifier i.
- *public_key:* Variables of this type represent agents' public keys in a public-key cryptosystem. For example, given a public (respectively, private) key pu, its inverse private (respectively, public) key is obtained by inv_pu.
- *symmetric_key:* This type represents a key for a symmetric-key cryptosystem.
- *text: text* values are often used as nonces. These values can be used for messages. If RNa is of type *text (fresh)*, then RNa' will be a fresh value, which the intruder cannot guess.
- *nat: nat* type represents the natural numbers in non message contexts.
- *const:* This type represents constants.
- *hash_func:* It represents a cryptographic hash function. The base type function also represents functions on the space of messages. It is assumed that the intruder cannot invert hash functions (in essence, that they are one-way collision-resistant functions).

The space of legal messages are defined as the closure of the basic types. For a given message M and encryption key k, $\{M\}\_k$ refers to as the symmetric/public-key encryption. The associative "·" operator is always used for concatenations.

The "*played_by X*" declaration indicates that the agent named in variable X will play in a specific role. A knowledge declaration (generally in the top-level *Environment* role) is used to specify the intruder's initial knowledge. Immediate reaction transitions have the form $A = | > B$, which relate an event A and an action B, and it indicates that whenever we take a transition that is labeled in such a way as to make the event predicate A true, we must immediately (that is, simultaneously) execute action B. If a variable X needs to be permanently kept secret, the goal *secrecy_of X* is to be used in HLPSL. As a result, if X is ever obtained or derived by the intruder, a security violation will result. A detailed description of AVISPA and HLPSL can be found in [1, 2].

7.1 Specifying our scheme

In this section, we discuss in brief the specification of our scheme for the roles of the user U_i, the GWN and the sensor node SN_j, the session, and also the goal and environment.

We have implemented the registration phase, the login phase and the authentication and key agreement phase of our scheme using the HLPSL language. In our implementation, we have three basic roles, namely *alice*, *server* and *bob*, which represent the participants as the user U_i, the GWN and the sensor node SN_j, respectively. The specification in HLPSL language for the role of the initiator, the user U_i is shown in Fig. 1. The user U_i first receives the start signal and changes its state from 0 to 1, and sends the registration request message $\langle ID_i, RPW_i \rangle$ securely to the GWN using the $Snd(\,)$ operation. The GWN then securely sends the smart card to the user U_i. During the login phase,

```
role alice (Ui, GWN, SNj : agent,
        H : hash_func,
        SKuigwn : symmetric_key,
        Snd, Rcv: channel(dy))
played_by Ui
def=
  local State  : nat,
    IDi, IDsnj, K, PWi, Bi, TS1, TS2, TS3, TS4: text,
    Kgwnu, Kgwns, Xs, TEi, TIDi, Ki, Kj : text,
    M1, RPWi, TCi, PTCi, PKSi, Ci, TIDinew : text
  const alice_server_ts1, server_alice_ts4,
      alice_server_tidi, server_alice_tidinew,
      subs1, subs2, subs3, subs4, subs5 : protocol_id
init  State := 0
transition
1. State   = 0 /\ Rcv(start) =|>
% Registration phase
State' := 1    /\ K' := new()
          /\ secret({PWi,Bi,K'},subs1,Ui)
          /\ RPWi' := H(IDi.K'.PWi)
          /\ Snd({IDi.RPWi'}_SKuigwn)
2. State = 1 /\ Rcv({H.TIDi'.TEi.xor(H(Kgwnu.IDi.TEi),
         H(IDi.K'.PWi)).H(IDi.Xs)}_SKuigwn)   =|>
% Login phase
State' := 2  /\ TS1' := new()
         /\ secret({Kgwnu,Kgwns,Xs,TEi},subs3,GWN)
         /\ secret(Ki,subs4,{GWN,Ui,SNj})
         /\ secret(Kj,subs5,{GWN,Ui,SNj})
% Ui sends the login message to the GWN
         /\ Snd(TIDi'.H(IDi.Ki.H(Kgwnu.IDi.TEi).
           H(IDi.Xs).TIDi'.TS1').xor(Ki,H(H(Kgwnu.IDi.
           TEi).H(IDi.Xs).TS1')).TS1')
% Ui has freshly generated the value TS1 for GWN
         /\ witness(Ui,GWN,alice_server_ts1, TS1')
         /\ witness(Ui,GWN,alice_server_tidi,TIDi')
% Authentication and key agreement phase
% Ui receives the acknowledgment message from GWN
2. State = 2  /\ Rcv(IDsnj.TS3'.TS4'.xor(Kj,H(xor(Ki,
        H(IDi.Xs)).TS3')).xor(TIDinew', H(xor(Ki,
        H(IDi.Xs).TS3'.TS4'))). H(IDi.IDsnj.
        H(Kgwnu.IDi.TEi). xor(TIDinew', H(xor(Ki,
        H(IDi.Xs).TS3'.TS4'))).Kj.TS3'.TS4')) =|>
% Ui's acceptance of the value TS4 generated for Ui by GWN
State' := 3 /\ request(GWN, Ui, server_alice_ts4, TS4')
        /\ request(Ui, GWN, server_alice_tidinew, TIDinew')
end role
```

Fig. 1 Role specification for the user U_i of our scheme

U_i sends the message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ to the GWN via a public channel and then waits to receive the acknowledgment $\langle ID_{SN_j}, TS_3, TS_4, PKS_j, D_{GWN}, E_{GWN}\rangle$ from the GWN using the $Rcv()$ operation.

In Fig. 2, we have shown the specification for the role of the GWN in HLPSL. After receiving the registration request message $\langle ID_i, RPW_i\rangle$ securely from the user U_i, it issues back a smart card to the GWN securely. When the GWN receives the message $\langle TID_i, C_i, PKS_i, TS_1\rangle$ from the user U_i during the login phase, the GWN sends the message $\langle TS_2, TID_i, C_{GWN}, PKS_{GWN}\rangle$ to the accessed sensor node SN_j. After that the GWN receives the message $\langle ID_{SN_j}, TS_3, C_j, PKS_j\rangle$ from SN_j. Finally, GWN replies to U_i with the message $\langle ID_{SN_j}, TS_3, TS_4, PKS_j, D_{GWN}, E_{GWN}\rangle$. Similarly, the role specification for the sensor node SN_j is provided in Fig. 3. In these specifications, secret ({PWi,Bi, K}, subs1, Ui) declaration tells that PWi, Bi, K are kept to the user U_i only, which is characterized by the protocol id subs1. Similarly, we have also specified the secrecy goals for subs2, subs3, subs4 and subs5 protocol ids. In HLPSL, witness (SNj, GWN, bob_server_ts3, TS3’) tells that SN_j has freshly generated the value TS_3 for the GWN. The declaration: request(GWN, SNj, server_bob_ts2, TS2’) is meant for SN_j’s acceptance of the value TS_2, which was generated for SN_j by the GWN.

In Fig. 4, we have given the specification in HLPSL for the role of session. Figure 5 shows the role specification for the goal and environment. In the session segment, all

```
role server (Ui, GWN, SNj : agent,
        H : hash_func,
        SKuigwn : symmetric_key,
        Snd, Rcv: channel(dy))
played_by GWN
def=
  local State  : nat,
     IDi, IDsnj, K, PWi, Bi, TS1, TS2, TS3, TS4 :  text,
     Kgwnu, Kgwns, Xs, TEi, TIDi, Ki, Kj : text,
     M1, RPWi, TCi, PTCi, PKSi, Ci, TIDinew : text
  const server_bob_ts2, bob_server_ts3, alice_server_ts1,
        server_alice_ts4, server_alice_tidinew,
        subs1, subs2, subs3, subs4, subs5 : protocol_id
init  State := 0
transition
% Registration phase
1. State = 0 ∧ Rcv({IDi.H(IDi.K’.PWi)}_SKuigwn) =|>
   State’ := 1 ∧ secret({PWi,Bi,K’},subs1,Ui)
             ∧ TIDi’ := new()
             ∧ Snd({H.TIDi’.TEi.xor(H(Kgwnu.IDi.TEi),
             H(IDi.K’.PWi)).H(IDi.Xs)}_SKuigwn)
% Login phase: receive the login request message from Ui
2. State = 1 ∧ Rcv(TIDi’.H(IDi.Ki.H(Kgwnu.IDi.TEi).
              H(IDi.Xs).TIDi’.TS1’).xor(Ki,H(H(Kgwnu.IDi.
              TEi).H(IDi.Xs).TS1’)).TS1’) =|>
State’ := 2 ∧ TS2’ := new( )
        ∧ secret({IDi},subs2,{Ui,GWN})
        ∧ secret({Kgwnu,Kgwns,Xs,TEi},subs3,GWN)
        ∧ secret(Ki,subs4,{GWN,Ui,SNj})
        ∧ secret(Kj,subs5,{GWN,Ui,SNj})
% Authentication and key agreement phase
        ∧ Snd(TS2’.TIDi’. H(TIDi’.H(Kgwns.IDsnj).TS2’).
         xor(xor(Ki,H(IDi.Xs)),H(H(Kgwns.IDsnj).TS2’)))
% GWN has freshly generated the value TS2 for SNj
        ∧ witness(GWN,SNj,server_bob_ts2, TS2’)
% Receive the message from sensor node SNj
2. State = 2 ∧ Rcv(IDsnj.TS3’.H(Kj.TIDi’.IDsnj.TS3’).
         xor(Kj, H(xor(Ki,H(IDi.Xs)).TS3’))) =|>
State’ := 3 ∧ TS4’ := new()
        ∧ TIDinew’ := new()
% Send the message back to the user Ui
        ∧ Snd(IDsnj.TS3’.TS4’.xor(Kj,H(xor(Ki,
        H(IDi.Xs)).TS3’)).xor(TIDinew’, H(xor(Ki,
        H(IDi.Xs).TS3’.TS4’))). H(IDi.IDsnj.
        H(Kgwnu.IDi.TEi). xor(TIDinew’, H(xor(Ki,
        H(IDi.Xs).TS3’.TS4’))).Kj.TS3’.TS4’))
% GWN has freshly generated the value TS4 for Ui
        ∧ witness(GWN,SNj,server_alice_ts4, TS4’)
% GWN has freshly generated the value TIDinew for Ui
        ∧ witness(GWN,SNj,server_alice_tidinew, TIDinew’)
% GWN’s acceptance of the value TS1 generated for GWN by Ui
        ∧ request(Ui, GWN, alice_server_ts1, TS1)
% GWN’s acceptance of the value TS3 generated for GWN by SNj
        ∧ request(SNj, GWN, bob_server_ts3, TS3’)
        ∧ request(Ui, GWN, alice_server_tidi, TIDi)
end role
```

Fig. 2 Role specification for the GWN of our scheme

```
role bob (Ui, GWN, SNj : agent,
        H : hash_func,
        SKuigwn : symmetric_key,
        Snd, Rcv: channel(dy))
played_by SNj
def=
  local State  : nat,
    IDi, IDsnj, K, PWi, Bi, TS1, TS2, TS3, TS4 :  text,
    Kgwnu, Kgwns, Xs, TEi, TIDi, Ki, Kj, Cj, PKSj: text,
    M1, M3, RPWi, TCi, PTCi, PKSi, Ci, TIDinew : text
  const server_bob_ts2, bob_server_ts3, alice_server_ts1,
      subs1, subs2, subs3, subs4, subs5 : protocol_id
init  State := 0
transition
% Authentication and key agreement phase
% Receive the message from the GWN
1. State = 0 ∧ Rcv(TS2’.TIDi’. H(TIDi’.H(Kgwns.IDsnj).TS2’).
   xor(xor(Ki,H(IDi.Xs)),H(H(Kgwns.IDsnj).TS2’))) =|>
State’ := 1
        ∧ TS3’ := new()
        ∧ Ki’ := new()
        ∧ secret({PWi,Bi,K},subs1,Ui)
        ∧ secret({IDi},subs2,{Ui,GWN})
        ∧ secret({Kgwnu,Kgwns,Xs,TEi},subs3,GWN)
        ∧ secret(Ki’,subs4,{GWN,Ui,SNj})
        ∧ secret(Kj,subs5,{GWN,Ui,SNj})
% Send the message to the GWN
        ∧ Snd(IDsnj.TS3’.H(Kj.TIDi’.IDsnj.TS3’).
         xor(Kj, H(xor(Ki,H(IDi.Xs)).TS3’)))
% SNj has freshly generated the value TS3 for GWN
        ∧ witness(SNj,GWN,bob_server_ts3, TS3’)
% SNj’s acceptance of the value TS2 generated for SNj by GWN
        ∧ request(GWN, SNj, server_bob_ts2, TS2’)
end role
```

Fig. 3 Role specification for the accessed sensor node SN_j of our scheme

```
role session(Ui,GWN,SNj: agent,
        % H is hash function
        H : hash_func,
        SKuigwn: symmetric_key)
def=
 local  US, UR, SS, SR, VS, VR: channel (dy)
 composition
        alice(Ui, GWN, SNj, H, SKuigwn, US, UR)
     ∧ server(Ui, GWN, SNj, H, SKuigwn, SS, SR)
     ∧ bob(Ui, GWN, SNj, H, SKuigwn, VS, VR)
end role
```

Fig. 4 Role specification for the session of our scheme

```
role environment()
def=
 const ui, gwn, snj : agent,
       h : hash_func,
       skuigwn: symmetric_key,
       idi, idsnj, ts1, ts2,
       ts3, ts4, tidi, tidinew : text,
       alice_server_ts1, server_bob_ts2,
       bob_server_ts3, server_alice_ts4,
       alice_server_tidi, server_alice_tidinew,
       subs1, subs2, subs3, subs4, subs5 : protocol_id
intruder_knowledge ={idsnj,h}
 composition
session(ui, gwn, snj, h, skuigwn)
   /\session(ui, gwn, snj, h, skuigwn)
   /\ session(ui, gwn, snj, h, skuigwn)
end role

goal
% subs1 represents that the password PWi and personal
% biometrics Bi are kept secret to the user Ui only.
secrecy_of subs1
% subs2 represents that the idenity IDi is
% kept secret to both Ui and GWN.
secrecy_of subs2
% subs3 represents that Kgwn−u, Kgwn−s, Xs and TEi
% are kept secret to the GWN only.
secrecy_of subs3
secrecy_of subs4
secrecy_of subs5
% Ui (the smart card) generates a timestamp TS1.
% When the GWN receives TS1 from the message from Ui,
% the GWN authenticates Ui based on TS1.
% Similarly for other protocols ids.
  authentication_on alice_server_ts1
  authentication_on server_bob_ts2
  authentication_on bob_server_ts3
  authentication_on server_alice_ts4
  authentication_on alice_server_tidi
  authentication_on server_alice_tidinew
end goal
environment()
```

Fig. 5 Role specification for the goal and environment of our scheme

the basic roles including the roles for U_i, GWN and SN_j are instanced with concrete arguments. The top-level role, which is called the environment, defines in the specification of HLPSL. It contains the global constants and a composition of one or more sessions, where the intruder may play some roles as legitimate users. In HLPSL, the intruder also participates in the execution of protocol as a concrete session. The current version of HLPSL supports the standard authentication and secrecy goals. In our implementation, we have verified five secrecy goals and six authentications, which are given in Fig. 5.

7.2 Analysis of results

We have first simulated our scheme using the widely-accepted OFMC backend [4] for the execution tests and a bounded number of sessions model checking. For the replay attack checking, this back-end checks whether the

```
% OFMC
% Version of 2006/02/13
SUMMARY
 SAFE
DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
 /home/avispa/web−interface−computation/
./tempdir/workfilefIFj04.if
GOAL
 as_specified
BACKEND
 OFMC
COMMENTS
STATISTICS
 parseTime: 0.00s
 searchTime: 1.86s
 visitedNodes: 158 nodes
 depth: 6 plies
```

Fig. 6 The result of the analysis using OFMC backend of our scheme

legitimate agents can execute the specified protocol by performing a search of a passive intruder. The back-end then provides the intruder about the knowledge of some normal sessions between the legitimate agents. For the Dolev-Yao model check, OFMC checks if there is any man-in-the-middle attack possible by the intruder. We have simulated our scheme under the OFMC backend using the AVISPA web tool [3]. The simulation results for the formal security verification shown in Fig. 6 clearly demonstrates that our scheme is secure against active attacks such as replay and man-in-the-middle attacks.

We have then simulated our scheme using CL-AtSe backend for the formal security verification under the AVISPA web tool [3]. The simulation results for the formal security

```
SUMMARY
 SAFE
DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
 TYPED_MODEL
PROTOCOL
 /home/avispa/web−interface−computation/
 ./tempdir/workfilefIFj04.if
GOAL
 As Specified
BACKEND
 CL−AtSe
STATISTICS
 Analysed   : 6 states
 Reachable  : 1 states
 Translation: 0.39 seconds
 Computation: 0.01 seconds
```

Fig. 7 The result of the analysis using CL-AtSe backend of our scheme

Table 5 Features comparison between our scheme and other schemes

Security features	[13]	[41]	[35]	[40]	[21]	Ours
SF_1	No	Yes	Yes	No	No	Yes
SF_2	Yes	Yes	Yes	Yes	Yes	Yes
SF_3	Yes	Yes	Yes	Yes	Yes	Yes
SF_4	No	No	Yes	No	Yes	Yes
SF_5	Yes	Yes	Yes	Yes	Yes	Yes
SF_6	Yes	Yes	Yes	Yes	Yes	Yes
SF_7	No	No	No	No	Yes	Yes
SF_8	No	No	No	No	Yes	Yes
SF_9	No	Yes	No	Yes	Yes	Yes
SF_{10}	No	Yes	Yes	Yes	Yes	Yes
SF_{11}	No	No	No	No	Yes	Yes
SF_{12}	No	No	No	No	Yes	Yes
SF_{13}	No	Yes	Yes	Yes	Yes	Yes
SF_{14}	No	Yes	No	No	No	Yes
SF_{15}	No	No	No	No	No	Yes
SF_{16}	No	No	No	No	No	Yes
SF_{17}	No	Yes	Yes	Yes	Yes	Yes
SF_{18}	No	No	No	No	No	Yes
SF_{19}	No	No	No	No	No	Yes
SF_{20}	No	No	No	No	No	Yes

Note: SF_1 : whether resilient against privileged insider attack; SF_2 : whether resilient against stolen-verifier attack; SF_3 : whether protects password guessing attack; SF_4 : whether resilient against stolen smart card attack; SF_5 : whether prevents forgery attack; SF_6 : whether resists replay attack; SF_7 : whether resilient against user identity guessing attack; SF_8 : whether resilient against tracing attack; SF_9 : whether provides mutual authentication between U_i and GWN; SF_{10} : whether provides mutual authentication between GWN and SN_j; SF_{11} : whether provides user anonymity; SF_{12} : whether provides user untraceability property; SF_{13} : whether supports key agreement between U_i and SN_j; SF_{14} : whether supports correct password update; SF_{15} : whether supports correct biometric update; SF_{16} : whether provides non-repudiation; SF_{17} : whether resilient against node capture attack; SF_{18} : whether provides three-factor security; SF_{19} : whether provides formal security analysis and verification; SF_{20} : whether supports dynamic sensor node addition after initial deployment.

verification shown in Fig. 7 clearly demonstrates that our scheme is also secure against active attacks such as replay and man-in-the-middle attacks.

8 Performance comparison with related schemes

In this section, we compare the features, computational overhead and communication overhead of our proposed scheme with those for other schemes such as M. L. Das's scheme [13], Yoo et al.'s scheme [41], Sun et al.'s scheme [35], Xue et al.'s scheme [40], Jiang et al.'s scheme [21].

8.1 Features comparison

In Table 5, we have summarized the comparison of security features provided by our scheme and other schemes. From this table, it is clear that our scheme provides better security features and higher security level. Our scheme has the ability to supports other good features such as correct password change by a legal user without contacting the GWN, biometric update by a legal user without contacting the GWN, non-repudiation because of applying biometric in our scheme, dynamic sensor node addition after initial deployment and formal security analysis and verification. Thus, our scheme is superior in terms of features as compared to those for other schemes.

8.2 Computational overhead comparison

In Table 6, we have compared our scheme for the computational cost with other schemes for the login phase and the authentication and key agreement phase. We denote t_h and t_{fe} for the time taken to perform one hashing operation and fuzzy extractor operation ($Gen(\cdot)$ or $Rep(\cdot)$), respectively. Note that the fuzzy extractor function is efficient. Due to efficiency of one-way hash function $h(\cdot)$, the computational cost of our scheme is comparable with other schemes. The rough estimation of computation cost required for the user registration, login and authentication phases of our scheme and other scheme, we use the simulated values of the execution time of one-way hash function $h(\cdot)$ and an elliptic curve point multiplication. As pointed out in [19], the computational time of a one-way hashing operation $h(\cdot)$ and an elliptic curve point multiplication are 0.00032 seconds and 0.0171 seconds, respectively. Also, as in [19], we assume that the time for executing a fuzzy extractor is same as that for an elliptic curve point multiplication at the most. Thus, we have, $t_h \approx 0.00032$ s and $t_{fe} \approx 0.0171$ s. From Table 6, we see that M. L. Das's scheme, Yoo et al.'s scheme, Sun et al.'s scheme, Xue et al.'s scheme, Jiang et al.'s scheme and our scheme require 0.00384 s, 0.00608 s, 0.00352 s, 0.01120 s, 0.00768 s and 0.04412 s, respectively, for all the user registration, login and authentication phases. However, for M. L. Das's scheme, Yoo et al.'s scheme, Sun et al.'s scheme, Xue et al.'s scheme, Jiang et al.'s scheme and our scheme, a sensor node SN_j requires the computation cost during the login and authentication phases as t_h (0.00032 s), $2t_h$ (0.00064 s), $2t_h$ (0.00064 s), $6t_h$ (0.00192 s), $5t_h$ (0.00160 s) and $5t_h$ (0.00160 s), respectively. Our scheme

Table 6 Computational overhead comparison between our scheme and other schemes

Phase	Entity	M. L. Das [13]	Yoo et al. [41]	Sun et al. [35]	Xue et al. [40]	Jiang et al. [21]	Ours
User registration	U_i	–	t_h	–	$2t_h$	t_h	$4t_h + t_{fe}$
	GWN	$3t_h$	$3t_h$	$2t_h$	$4t_h$	t_h	$2t_h$
Login +	U_i	$4t_h$	$5t_h$	$2t_h$	$10t_h$	$7t_h$	$t_{fe} + 9t_h$
Authentication	GWN	$4t_h$	$8t_h$	$5t_h$	$13t_h$	$10t_h$	$11t_h$
	SN_j	t_h	$2t_h$	$2t_h$	$6t_h$	$5t_h$	$5t_h$
Total cost		$12t_h$	$19t_h$	$11t_h$	$35t_h$	$24t_h$	$31t_h + 2t_{fe}$
Rough estimation		0.00384 s	0.00608 s	0.00352 s	0.01120 s	0.00768 s	0.04412 s

is also much suited for the resource-constrained sensor nodes due to computational efficiency of $h(\cdot)$. Again, the user registration process is a one-time procedure. Though our scheme requires little more computational cost due to fuzzy extractor operations than other schemes during the login and authentication phases, our scheme provides better security features and higher security level than other schemes.

8.3 Communication overhead comparison

Finally, in Table 7, we have compared the communication cost of our scheme with other schemes for the login phase and the authentication and key agreement phase. We assume that the hash digest (output) is 160 bits, if we use SHA-1 hash function [33], timestamp TS is 32 bits, expiration time TE_i for a temporal credential is 32 bits, user identity ID_i is 160 bits, user temporary identity TID_i is 160 bits, random nonce is 160 bits and sensor node identity ID_{SN_j} is 16 bits. It is clear that Xue et al.'s scheme [40] needs more communication cost compared to all schemes. Though M. L. Das's scheme [13] requires minimum communication cost among all the schemes, it fails to provide several desired security features, such as mutual authentication, session key agreement, password and biometric update, non-repudiation, and formal security verification. Further, our scheme requires little more communication cost as compared to Jiang et al.'s scheme [21]. This is justified because our scheme provides better security features and higher security level than other schemes.

Table 7 Communication overhead comparison between our scheme and other schemes

Scheme	Communication overhead
M. L. Das [13]	2 messages (704 bits)
Yoo et al. [41]	6 messages (1824 bits)
Sun et al. [35]	5 messages (1296 bits)
Xue et al. [40]	4 messages (2256 bits)
Jiang et al. [21]	4 messages (1920 bits)
Ours	4 messages (1952 bits)

9 Conclusion

In this paper, we have first reviewed the recently proposed Jiang et al.'s password-based user authentication scheme for WSNs using smart card. We have then pointed out that their scheme has several drawbacks. In order to withstand these drawbacks found in Jiang et al.'s scheme, we have proposed an improved scheme using biometric, password and smart card of a legal user, while retaining the original merits of Jiang et al.'s scheme. We have shown that our scheme provides better security features and higher security level than other schemes, which are demonstrated through the formal and informal security analysis. In addition, we have performed the simulation for the formal security verification of our scheme using the widely-accepted AVISPA tool and the simulation results confirm that our scheme is also secure against replay and man-in-the-middle attacks. Our scheme is also efficient in terms of communication and computational overheads as compared to other schemes. Overall, considering better security features and higher security level, and efficiency that our scheme provides, we conclude that our scheme is more appropriate for practical applications such as healthcare and battlefield applications of WSNs as compared to other existing approaches.

Acknowledgments The authors would like to acknowledge the many helpful suggestions of the anonymous reviewers and the Editor, which have improved the content and the presentation of this paper.

Conflict of interests The author declares that there is no conflict of interest.

References

1. Armando A (2005) The AVISPA tool for the automated validation of internet security protocols and applications. In: 17th International conference on computer aided verification (CAV'05). (Lecture Notes in Computer Science), vol 3576. Springer, Berlin, pp 281–285
2. AVISPA Automated Validation of Internet Security Protocols and Applications. http://www.avispa-project.org/. Accessed on January 2013.
3. AVISPA AVISPA web tool. http://www.avispa-project.org/web-interface/expert.php/. Accessed on July 2014
4. Basin D, Modersheim S, Vigano L (2005) OFMC: A symbolic model checker for security protocols. Int J Inf Secur 4(3): 181–208
5. Burnett A, Byrne F, Dowling T, Duffy A (2007) A Biometric Identity Based Signature Scheme. Int J Inf Secur 5(3):317–326
6. Chen TH, Shih WK (2010) A robust mutual authentication protocol for wireless sensor networks. ETRI J 32(5):704–712
7. Chuang YH, Tseng YM (2010) An efficient dynamic group key agreement protocol for imbalanced wireless networks. Int J Netw Manag 20(4):167–180
8. Das AK (2011) Analysis and improvement on an efficient biometric-based remote user authentication scheme using smart cards. IET Inf Secur 5(3):145–151
9. Das AK, Chatterjee S, Sing JK (2014) Formal security analysis and verification of a password-based user authentication scheme for hierarchical wireless sensor networks. Int J Trust Manag Comput Commun (Inderscience) 2(1):78–102
10. Das AK, Goswami A (2013) A secure and efficient uniqueness-and-anonymity-preserving remote user authentication scheme for connected health care. J Med Syst 37(3):1–16
11. Das AK, Paul NR, Tripathy L (2012) Cryptanalysis and improvement of an access control in user hierarchy based on elliptic curve cryptosystem. Inf Sci 209(C):80–92
12. Das AK, Sharma P, Chatterjee S, Sing JK (2012) A dynamic password-based user authentication scheme for hierarchical wireless sensor networks. J Netw Comput Appl 35(5):1646–1656
13. Das ML (2009) Two-factor user authentication in wireless sensor networks. IEEE Trans Wirel Commun 8(3):1086–1090
14. Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–654
15. Dodis Y, Reyzin L, Smith A (2004) Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Proceedings of the advances in cryptology (Eurocrypt'04), LNCS vol 3027. pp 523–540
16. Dolev D, Yao A (1983) On the security of public key protocols. IEEE Trans Inf Theory 29(2):198–208
17. Fan R, Ping LD, Fu JQ, Pan XZ (2010) A secure and efficient user authentication protocol for two-tieres wireless sensor networks. In: 2nd pacific-asia conference on circuits, communications and system (PACCS 2010). pp 425–428
18. He D, Gao Y, Chan S, Chen C, Bu J (2010) An enhanced two-factor user authentication scheme in wireless sensor networks. Ad hoc & sensor wireless networks 10(4)
19. He D, Kumar N, Lee JH, Sherratt RS (2014) Enhanced three-factor security protocol for consumer USB mass storage devices. IEEE Trans Consum Electron 60(1):30–37
20. Huang HF, Chang YF, Liu CH (2010) Enhancement of two-factor user authentication in wireless sensor networks. In: 6th international conference on intelligent information hiding and multimedia signal processing. pp 27–30
21. Jiang Q, Ma J, Lu X, Tian Y (2014) An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks
22. Khan MK, Alghathbar K (2010) Cryptanalysis and security improvements of two-factor user authentication in wireless sensor networks. Sensors 10(3):2450–2459
23. Khan MK, Zhang J, Wang X (2008) Chaotic hash-based fingerprint biometric remote user authentication scheme on mobile devices. Chaos, Solitons Fractals 35(3):519–524
24. Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Proceedings of advances in cryptology - CRYPTO'99, LNCS, vol. 1666. pp 388–397
25. Li CT, Hwang MS (2010) An efficient biometric-based remote authentication scheme using smart cards. J Netw Comput Appl 33(1):1–5
26. Li X, Niu JW, Ma J, Wang WD, Liu CL (2011) Cryptanalysis and improvement of a biometrics-based remote user authentication scheme using smart cards. J Netw Comput Appl 34(1): 73–79
27. Messerges TS, Dabbish EA, Sloan RH (2002) Examining smart-card security under the threat of power analysis attacks. IEEE Trans Comput 51(5):541–552
28. Nyang D, Lee MK (2009) Improvement of Das's two-factor authentication. protocol in wireless sensor networks. http://eprint.iacr.org/2009. Report 2009/631
29. Odelu V, Das AK, Goswami A (2014) A secure effective key management scheme for dynamic access control in a large leaf class hierarchy. Inf Sci 269(C):270–285
30. von Oheimb D (2005) The high-level protocol specification language hlpsl developed in the eu project avispa. In: Proceedings of APPSEM 2005 workshop
31. Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
32. Sarkar P (2010) A simple and generic construction of authenticated encryption with associated data. ACM Trans Inf Syst Secur 13(4):33
33. Secure Hash Standard FIPS PUB 180-1, National institute of standards and technology (nist), u.s. department of commerce, April 1995
34. Stinson DR (2006) Some observations on the theory of cryptographic hash functions. Des Codes Crypt 38(2):259–277
35. Sun DZ, Li JX, Feng ZY, Cao ZF, Xu GQ (2013) On the security and improvement of a two-factor user authentication scheme in wireless sensor networks. Pers Ubiquit Comput 17(5): 895–905
36. Tan Z (2014) A user anonymity preserving three-factor authentication scheme for telecare medicine information systems. J Med Syst 38(3):1–9
37. Vaidya B, Makrakis D, Mouftah HT (2010) Improved two-factor user authentication in wireless sensor networks. In: 2nd international workshop on network assurance and security services in ubiquitous environments. pp 600–606

38. Watro R, Kong D, Cuti S, Gardiner C, Lynn C, Kruus P (2004) TinyPK: Securing sensor networks with public key technology. In: Proceedings of the 2nd ACM workshop on security of ad hoc and sensor networks, SASN 2004. USA, Washington, DC, pp 59–64
39. Wong K, Zheng Y, Cao J, Wang S (2006) A dynamic user authentication scheme for wireless sensor networks. In: Proceedings of IEEE international conference sensor networks, ubiquitous, trustworthy computing, IEEE Computer Society, pp 244–251
40. Xue K, Ma C, Hong P, Ding R (2013) A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. J Netw Comput Appl 36(1):316–323
41. Yoo SG, Park KY, Kim J (2012) A security-performance-balanced user authentication scheme for wireless sensor networks. In: International journal of distributed sensor networks 2012 (2012). Article ID 382810, 11 pages. doi:10.1155/2012/382810
42. Yuan J, Jiang C, Jiang Z (2010) A biometric-based user authentication for wireless sensor networks. Wuhan Univ J Nat Sci 15(3):272–276

Ashok Kumar Das is currently working as an Assistant Professor in the Center for Security, Theory and Algorithmic Research of the International Institute of Information Technology (IIIT), Hyderabad 500 032, India. He received his Ph.D. degree in Computer Science and Engineering, M.Tech. degree in Computer Science and Data Processing, and M.Sc. degree in Mathematics, all from the Indian Institute of Technology, Kharagpur, India. He received the Institute Silver Medal from the Indian Institute of Technology, Kharagpur, India. His current research interests include cryptography, wireless sensor network security, proxy signature, hierarchical access control, data mining and remote user authentication. He has published more than 60 papers in international journals and conferences in these areas. For more details, please visit https://sites.google.com/site/iitkgpakdas/, http://www.iiit.ac.in/people/faculty/ashokkdas.