INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. (2015)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/dac.2933

A secure and effective biometric-based user authentication scheme for wireless sensor networks using smart card and fuzzy extractor

Ashok Kumar Das*,†

Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India

SUMMARY

User authentication is a prominent security requirement in wireless sensor networks (WSNs) for accessing the real-time data from the sensors directly by a legitimate user (external party). Several user authentication schemes are proposed in the literature. However, most of them are either vulnerable to different known attacks or they are inefficient. Recently, Althobaiti *et al.* presented a biometric-based user authentication scheme for WSNs. Although their scheme is efficient in computation, in this paper, we first show that their scheme has several security pitfalls such as (i) it is not resilient against node capture attack; (ii) it is insecure against impersonation attack; and (iii) it is insecure against man-in-the-middle attack. We then aim to propose a novel biometric-based user authentication scheme suitable for WSNs in order to withstand the security pitfalls found in Althobaiti *et al.* scheme. We show through the rigorous security analysis that our scheme is secure and satisfies the desirable security requirements. Furthermore, the simulation results for the formal security verification using the most widely used and accepted Automated Validation of Internet Security Protocols and Applications tool indicate that our scheme is secure. Our scheme is also efficient compared with existing related schemes. Copyright © 2015 John Wiley & Sons, Ltd.

Received 3 September 2014; Revised 27 December 2014; Accepted 7 January 2015

KEY WORDS: wireless sensor networks; user authentication; smart cards; biometrics; cryptanalysis; AVISPA; security.

1. INTRODUCTION

A wireless sensor network (WSN) is considered as a large network having several tiny computing nodes, also called the sensors or motes. They can be deployed in a deployment field or target field randomly or in a group. These nodes have the ability to sense important observations from their surrounding areas and then transmit those sensing data to the nearby base stations (BSs), which do further processing on behalf of them. Sensor nodes can communicate among each other via short-range radio wireless communications. The BS (known as the gateway node (GWN or GW-node)) is the most powerful node in WSN. The sensors are extremely resource-starved, which have lack of memory, computational capability, and radio transmission range.

Sensor networks are widely used in various applications starting from military to environmental and medical research in recent years [1–9]. In many WSN applications including target tracking, battlefield surveillance, and intruder detection, WSNs often operate in unattended environment. Thus, an adversary has an opportunity to directly capture a sensor node from the target field and extract all the information from its memory as nodes are not generally tamper-resistant due to their cost effectiveness. Hence, there is a strong need for protecting the sensing data and sensing readings in WSNs. In wireless communications, an adversary not only can eavesdrop the traffic but also

*Correspondence to: Ashok Kumar Das, Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India.
†E-mail: iitkgp.akdas@gmail.com

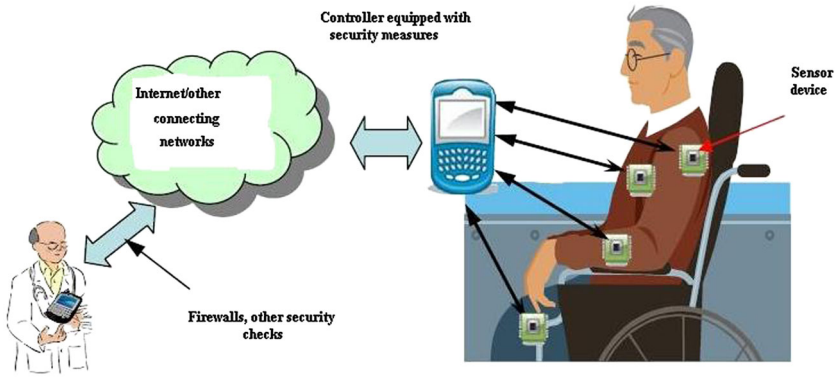

Figure 1. Security measures at gateway in wireless body area networks (Source: [15]).

can intercept or interrupt the exchanged messages. As a result, several protocols and algorithms designed for the traditional networks are not suitable for hostile environments unless we take care of the adequate security measures. Hence, security in WSNs becomes an important concern as there are several potential attacks against sensor networks. The readers can find the surveys on WSNs and their security issues in [10–14].

Consider an example of critical applications in medical healthcare system [15] shown in Figure 1. In a wireless body area network, sensor nodes are placed in a patient's body for measuring medical data such as ECG, body movement, temperature, respiration, heart rate, pulse oximeter, blood pressure, and blood sugar. The deployed sensors then transmit sensing data to a small body area network gateway. The gateway processes data locally and resends them to the router for the external network to the medical server at the hospital. The results can be then observed and analyzed by the medical staffs/doctors to monitor patients and take the correct immediate decisions, if needed. Because the patient's data are confidential, only authorized users must be given access to monitor the patient. To obtain the real-time information from a patient's body, the external users (in this example, the doctors in the hospital) must be authenticated by the BS (medical server) and the sensors before allowing access to the sensing data from the sensors inside wireless body area network.

We also consider another military application of the battlefield surveillance [16]. Several sensors are rapidly scattered in a target field with the help of airplanes or trucks. The deployed sensors then monitor information from their surrounding areas and send the sensing information to the nearby BS through their neighboring nodes. After that, the BS performs a more accurate detection on the activities such as possible attacks of the opposing force. Hence, if the real-time data are available at that time, the decisions and responses can be taken quickly.

We now consider the necessity of a user authentication problem in WSNs as follows. Several critical applications in WSNs including the applications of the battlefield surveillance and medical healthcare system discussed earlier are real-time-based, and the users usually access the real-time data from the sensors inside WSNs, because the data available at the BS may not be always real-time and they are gathered periodically by the BS from the nodes in WSNs [16]. To access the real-time data from the nodes, the user needs to be first authenticated to the nodes and the BS and then needs to establish a secret session key between the user and the accessed node so that the illegal access does not occur. Because of this reason, the user authentication problem is a very important research topic in WSN security, which has received considerable research attention in WSN security study in the recent years.

Several user authentication schemes using passwords are presented in the literature [17–23]. However, most of these schemes are insecure against various known attacks. Das *et al.* [16] proposed a novel and efficient password-based user authentication scheme for the hierarchical WSNs. Their scheme was shown to be secure against various known attacks including the replay and man-in-the-middle attacks with the help of formal security verification [24]. Further, an improved version of Das *et al.* scheme [16] has been proposed in [25] in the literature. User authentication using biometrics in WSNs has become popular because of its more reliability and security compared with

DOI: 10.1002/dac

the traditional password-based user authentication schemes in WSNs. Yuan *et al.* scheme [26] has better security as compared with that for [27]. However, Yuan *et al.* scheme [26] has similar drawbacks as in [27], such as it is insecure against denial-of-service and node capture attacks. Das *et al.* introduced a new secure biometric-based user authentication scheme in hierarchical wireless body area sensor networks [28]. Althobaiti *et al.* [29] further presented an efficient biometric-based user authentication scheme for WSNs. Unfortunately, it is shown that their scheme has several security pitfalls [30], and as a result, their scheme is not practical to use for the real-life WSN applications. Das and Bezawada [31] proposed a user authentication mechanism in heterogeneous WSNs based on biometrics, passwords, and smart cards, which can defend various known attacks. Furthermore, although it is efficient, it lacks support of the dynamic node addition. Das [32] proposed another secure and robust temporal credential-based three-factor user authentication approach for WSNs, which requires little more computational effort as compared with our scheme proposed in this paper. He *et al.* [33] also proposed a temporal–credential-based scheme for WSNs, which provides both mutual authentication and key agreement. However, their scheme does not support dynamic node addition feature. Their scheme is based on the public-key cryptographic technique and requires computation of modular exponentiation operation, which is expensive for the resource-constrained sensor nodes. In addition, their scheme lacks the formal security verification.

In this paper, a new biometric-based authentication scheme for WSNs using smart card is proposed. We show that our scheme satisfies the desirable security requirements needed for user authentication in WSNs. Further, we show that our scheme is also comparable with other existing schemes for the communication and computation overheads. Using the simulation through the widely accepted Automated Validation of Internet Security Protocols and Applications (AVISPA) tool for the formal security verification, we further show that our scheme is secure.

The organization of the remainder of the paper is as follows. In Section 2, we discuss some basic mathematical preliminaries, which are needed for reviewing and cryptanalysis of Althobaiti *et al.* scheme [29] and also for describing and analyzing our proposed scheme. In Section 3, we describe the recently proposed Althobaiti *et al.* scheme. We then show that Althobaiti *et al.* scheme is insecure against several attacks in Section 4. In Section 5, we describe our proposed new user authentication scheme for WSNs. In Section 6, we perform the security analysis of our scheme. We provide the simulation of our scheme for the formal security verification with the help of the AVISPA tool in Section 7. We compare the performance and functionality of our scheme with those of other existing schemes in Section 8. Finally, Section 9 concludes the paper with some concluding remarks.

2. SOME MATHEMATICAL PRELIMINARIES

In this section, we briefly describe the following basic mathematical preliminaries, which are essential for describing and analyzing Althobaiti *et al.* scheme [29] and our scheme.

2.1. Collision-resistant one-way hash function

This function can be formally defined as follows [34–36].

Definition 1
A collision-resistant one-way hash function is a many-one mapping $h : A \rightarrow B$, where $A = \{0, 1\}^*$ and $B = \{0, 1\}^n$. It is a deterministic algorithm, which takes an arbitrary-length binary string, say $x \in A$ as input and then gives a binary string of fixed-length, n, say $y \in B$ as output. An adversary (attacker) $\mathcal{A}$'s advantage in finding collision is defined by

$$Adv_{\mathcal{A}}^{HASH}(t) = Pr[(x, x') \leftarrow_R \mathcal{A} : x \neq x', h(x) = h(x')], \tag{1}$$

where $Pr[E]$ denotes the probability of a random event E, and $(x, x') \leftarrow_R \mathcal{A}$ means that the pair (x, x') is selected randomly by $\mathcal{A}$. $\mathcal{A}$ is allowed to be probabilistic. In this case, the probability in the advantage is computed over the random choices made by $\mathcal{A}$ with execution time t. The hash function $h(\cdot)$ is collision-resistant, if $Adv_{\mathcal{A}}^{HASH}(t) \leqslant \tau$, for any sufficiently small $\tau > 0$.

DOI: 10.1002/dac

2.2. Indistinguishability of encryption and chosen plaintext attack

The indistinguishability of encryption and chosen plaintext attack (IND-CPA) is formally given as follows [37, 38].

Definition 2
Let *SA*/*MA* denote a single/multiple eavesdropper. Let $O_{sk_1}, O_{sk_2}, \ldots, O_{sk_N}$ be N different encryption oracles associated with the secret keys, say $sk_1, sk_2, \ldots, sk_N$, respectively. The advantage functions of *SA* and *MA* are $Adv^{IND\text{-}CPA}_{\Omega,SE}(l) = 2Pr[SA \leftarrow O_{sk_1}; (b_0, b_1 \leftarrow_R SA); \tau \leftarrow_R \{0,1\}; \gamma \leftarrow_R O_{sk_1}(b_\tau) : SE(\gamma) = \tau] - 1$, and $Adv^{IND\text{-}CPA}_{\Omega,MA}(l) = 2Pr[MA \leftarrow O_{sk_1}, O_{sk_2}, \ldots, O_{sk_N}; (b_0, b_1 \leftarrow_R MA); \tau \leftarrow_R \{0,1\}; \gamma_1 \leftarrow_R O_{sk_1}(b_\tau), \gamma_2 \leftarrow_R O_{sk_2}(b_\tau), \ldots, \gamma_N \leftarrow_R O_{sk_N}(b_\tau) : MA(\gamma_1, \gamma_2, \ldots, \gamma_N) = \tau] - 1$, respectively. We denote $\tau \leftarrow_R \{0,1\}$ that τ is a randomly chosen bit from the binary set $\{0,1\}$. The encryption scheme, say Ω, is said to be IND-CPA secure if $Adv^{IND\text{-}CPA}_{\Omega,SA}(l)$ (resp. $Adv^{IND\text{-}CPA}_{\Omega,MA}(l)$) is negligible for any probabilistic, polynomial time adversary *SA* (respectively, *MA*).

2.3. 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 [39–41] is preferred, which can extract a uniformly random string and a public information from the biometric template with a given error tolerance $\mathcal{T}$. In the reproduction process, the fuzzy extractor recovers the original biometric key data for a noisy biometric using public information and $\mathcal{T}$. Let $\mathcal{M} = \{0,1\}^v$ be a finite v-dimensional metric space of biometric data points, $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{Z}^+$ a distance function useful for computing the distance between any two points based on the chosen metric, l the bit-length of the output string, and $\mathcal{T}$ the error tolerance parameter, where $\mathbb{Z}^+$ is the set of all positive integers.

Definition 3
The fuzzy extractor is a tuple $(\mathcal{M}, l, \mathcal{T})$, which is composed of the following two algorithms, called *Gen* and *Rep*

- *Gen.* This probabilistic algorithm 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 deterministic algorithm takes a noisy biometric information $B_i' \in \mathcal{M}$ and a public parameter τ_i and $\mathcal{T}$ 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 $d(B_i, B_i') \leqslant \mathcal{T}$ is satisfied.

One can refer to [39, 40] for details of the fuzzy extractor and the extraction process.

3. REVIEW OF ALTHOBAITI *ET AL.* SCHEME

In this section, we briefly review the recently proposed Althobaiti *et al.* biometric-based user authentication scheme in WSNs [29]. Various phases of their scheme are discussed in the following subsections. Notations in Table I are used for describing and analyzing Althobaiti *et al.* scheme.

3.1. Registration phase

For the registration of a user U_i, the system randomly selects an encryption key, say ek_i, and it is saved in the GWN as a secret key of U_i. The cryptographic hash function $h(\cdot)$ (for example, SHA256 [42]) is applied on the extracted features of U_i's biometric (for example, iris), and the produced hash value is XORed with the encryption key ek_i to generate BE template, which is stored in U_i's

DOI: 10.1002/dac

Table I. Notations used.

Symbol	Explanation
GWN	WSN gateway node (GW-node or 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
X_s	1024-bit secret master key of GWN
SN_j	j^{th} sensor node in WSN
ID_{SN_j}	Identity of SN_j
X	Secret information shared by GWN and all deployed sensor nodes
$E_k(\cdot)$	Symmetric encryption using the key k
$D_k(\cdot)$	Symmetric decryption using k
$MAC_k(m)$	Message authentication code of m using key k
$h(\cdot)$	Secure one-way hashing function
$Gen(\cdot)$	Fuzzy generator function
$Rep(\cdot)$	Fuzzy reproduction function
$\mathcal{T}$	Error tolerance threshold used in fuzzy extractor
ΔT	Maximum transmission delay
$A\|\|B$	Concatenation of A and B
$A \oplus B$	Bitwise XORed of A and B

device. U_i's data (identity ID_i, name, etc.) and also ek_i are saved in the GWN's database. The GWN computes $F_i = h(ID_i \oplus X)$, where X is a secret GWN's generated parameter, which is also saved in all the sensors SN_j before their deployment in a particular target field. Finally, the GWN sends the registration message $\langle ID_i, F_i \rangle$ to U_i via a secure channel. In this scheme, as in [27], all the deployed sensor nodes SN_j know the secret parameter X, and they are also responsible to respond to the data/query asked by U_i. Note that U_i's device contains the information $\{ID_i, F_i, h(ek_i), BE\}$, where $BE = h(biometric\_feature) \oplus ek_i$.

3.2. Login phase

The following steps are executed in this phase.

Step 1. A legal user U_i first inputs his/her identity ID_i and then inputs the personal biometric, iris, with the help of camera. The biometric features of U_i's iris are extracted, corrected by error correcting code, and also hashed by SHA256 hashing algorithm.
Step 2. The encryption key is re-computed as $ek_i' = BE \oplus h(biometric\_feature)$.
Step 3. ek_i' is hashed, and after that, $h(ek_i)$ stored in the device is compared with the computed $h(ek_i')$. If there is a match, a login request $\langle ID_i, request \rangle$ is sent to the GWN along with ID_i via a public channel. Otherwise, the session is terminated immediately.

3.3. Authentication phase

It works as follows.

Step 1. When the login request from U_i is received, the GWN replies to U_i with the authentication request $\langle R \rangle$, where R is a random challenge. When U_i receives the message from the GWN, U_i encrypts R and T_1 with the encryption key ek_i derived in the login phase, where T_1 is the current timestamp of U_i's device and sends the authentication request message $\langle E_{ek_i}(R, T_1) \rangle$ to the GWN via a public channel.
Step 2. After receiving the authentication request message from U_i, the GWN decrypts the message using the encryption key ek_i stored in the GWN and checks the condition $|T_1 - T_2| < \Delta T$, where ΔT is the maximum transmission delay and T_2 the time

DOI: 10.1002/dac

Table II. Summary of exchanged messages during login and authentication phases of Althobaiti *et al.* scheme.

User U_i	GWN	Sensor SN_j
Login phase		
$\langle ID_i, request \rangle \xrightarrow{\quad\quad}$		
Authentication phase		
	$\xleftarrow{\quad} \langle R \rangle$	
$\langle E_{ek_i}(R, T_1) \rangle \xrightarrow{\quad\quad}$		
	$\langle ID_i, Y_i, T_3 \rangle \xrightarrow{\quad\quad}$	
Receives $\langle L, T_5 \rangle$ from SN_j		$\xleftarrow{\quad} \langle L, T_5 \rangle$

when the message was received. If this condition is invalid, the authentication phase is terminated immediately.

Let the query made by U_i be replied by a login-sensor node SN_j. The GWN computes $F_i = h(ID_i \oplus X)$ and $Y_i = MAC_{F_i}(ID_i||SN_j||T_3)$, where T_3 is the GWN's current timestamp. The GWN then sends the message $\langle ID_i, Y_i, T_3 \rangle$ to SN_j via a public channel.

Step 3. When SN_j receives the message from the GWN, SN_j checks the validity of T_3 by verifying the condition $|T_3 - T_4| < \Delta T$, where T_4 is the time when the message was received. If the condition is valid, SN_j computes $F_i = h(ID_i \oplus X)$, $Y_i' = MAC_{F_i}(ID_i||SN_j||T_3)$ and checks if $Y_i' = Y_i$. If it holds, SN_j responds to U_i's query (RM), computes $V_i = h(ID_i||F_i||T_5)$, $C_i = h(RM)$, and $L = E_{V_i}(RM, C_i)$, and then sends the message $\langle L, T_5 \rangle$ to U_i via a public channel, where T_5 is the SN_j's current timestamp.

Step 4. Finally, when U_i receives the message from SN_j at time T_6, U_i first validates it by checking whether $|T_5 - T_6| < \Delta T$, and if it is valid, then U_i computes $V_i = h(ID_i||F_i||T_5)$. After that, U_i decrypts L to retrieve RM and C_i as $(RM', C_i') = D_{V_i}(L)$ and then computes $C_i^* = h(RM')$. If $C_i^* = C_i'$, U_i accepts RM as a valid query response from SN_j. Otherwise, U_i rejects RM. Note that in this scheme, $V_i = h(ID_i||F_i||T_5)$ is considered as a session key between U_i and SN_j.

The login and authentication phases are briefly given in Table II.

4. CRYPTANALYSIS ON ALTHOBAITI *ET AL.* SCHEME

In this section, we first supply a threat model in Section 4.1 under which the security of WSN is generally evaluated. After that, it is shown that Althobaiti *et al.* scheme is insecure against several attacks, which are outlined in Section 4.2.

4.1. Threat model

For evaluating the security analysis of Althobaiti *et al.* scheme and also our proposed scheme, we use the threat model as follows. In most applications, sensors are usually deployed in the hostile environments. We assume that cost-effective sensor nodes are not tamper-resistant. Further, we assume that sensor nodes can be physically captured by an adversary either randomly or selectively from WSN, and all the sensitive data stored in their memory are known to that adversary. From the literature, it is known that even if the sensors are tamper-resistant, an adversary can still extract all the sensitive data stored in their memory by monitoring the power analysis attacks [43, 44]. However, in any case, the GWN will not be compromised by an adversary, because if the GWN is compromised, the entire network may be compromised. Thus, the GWN is trusted and it will not misuse the encryption keys of the legal users and the keys of sensor nodes shared between the GWN and them. As in [27], the Dolev–Yao threat model [45] is used in this paper, in which any two nodes communicate over a public channel. This model is suitable for WSNs as the channel is public and the end-points (sensor nodes) are not trusted. In addition, it is assumed that an adversary can intercept all traffic and inject packets and also reply the old previously transmitted packets.

DOI: 10.1002/dac

4.2. Attacks on Althobaiti et al. scheme

We show that Althobaiti *et al.* scheme is insecure against the following attacks.

4.2.1. Resilience against node capture attack. Suppose an adversary $\mathcal{A}$ captures c nodes either randomly or selectively from the target field. Then $\mathcal{A}$ knows all the information from their memory. Knowing these information, $\mathcal{A}$ can compromise a fraction of total secure communications that are compromised by c nodes *not including* the communication in which those nodes are directly involved [16]. Thus, this reflects on the effect of c sensor nodes being compromised on the rest of the network. If $P_e(c)$ is the probability that $\mathcal{A}$ can decrypt the secure communication between a non-compromised sensor node SN_j and a user U_i when c nodes are already compromised, and $P_e(c) = 0$, we say that a user authentication mechanism is *unconditionally secure against node capture attack*.

Suppose $\mathcal{A}$ captures a login-sensor node, say SN_j. Then $\mathcal{A}$ knows the secret parameter X stored in the sensor SN_j's memory and the GWN. Intercepting the messages $\langle ID_i, Y_i, T_3\rangle$ and $\langle L, T_5\rangle$ during the authentication phase, $\mathcal{A}$ can compute $F_i = h(ID_i \oplus X)$ and $V_i = h(ID_i||F_i||T_5)$, which is the session key between U_i and SN_j. Hence, $\mathcal{A}$ knows the session key V_i. We now show that $\mathcal{A}$ has the ability to compromise all the session keys between U_i and any other non-compromised sensor nodes SN'_j as follows. Let the GWN send the message $\langle ID_i, Y'_i, T'_3\rangle$ to the sensors SN'_j and SN'_j, which are non-compromised nodes, and send the message $\langle L', T'_5\rangle$ during the authentication phase, where $F_i = h(ID_i \oplus X)$, $Y'_i = MAC_{F_i}(ID_i||SN'_j||T'_3)$, $C'_i = h(RM)$, $V'_i = h(ID_i||F_i||T'_5)$ and $L' = E_{V'_i}(RM, C'_i)$. Because $\mathcal{A}$ knows X, ID_i, and T'_5, he/she can easily derive the session key $V'_i = h(ID_i||F_i||T'_5)$. It is then clear that $\mathcal{A}$ can derive all the session keys between U_i and any non-compromised sensor node SN'_j even if a single login-sensor node is already compromised in WSN. The effect of compromising of a single sensor node leads to compromise the successful decryptions of all secure communications between U_i and any non-compromised sensor SN'_j. Thus, we have $P_e(c) = 1.0$. Hence, Althobaiti *et al.* scheme is not at all resilient against node capture attack.

4.2.2. Impersonation attack. We show that an adversary $\mathcal{A}$ can impersonate the GWN to a login-sensor node. The detailed description is as follows. Suppose $\mathcal{A}$ physically captures a login-sensor node, say SN_j. $\mathcal{A}$ then knows the secret parameter X from the captured node SN_j. $\mathcal{A}$ also intercepts the message $\langle ID_i, Y_i, T_3\rangle$ during the authentication phase. Let $\mathcal{A}$ wish to impersonate the GWN to another non-compromised login-sensor node SN'_j. For this purpose, $\mathcal{A}$ can compute $F'_i = h(ID_i \oplus X)$ and $Y'_i = MAC_{F'_i}(ID_i||SN'_j||T'_3)$, where SN'_j denotes the sensor node from which U_i is expecting the response of the query, and T'_3 is the current timestamp of $\mathcal{A}$'s system. $\mathcal{A}$ then sends the message $\langle ID_i, Y'_i, T'_3\rangle$ to SN'_j via a public channel. After receiving the message, SN'_j checks the validity of T'_3. If it is valid, SN'_j computes $F_i = h(ID_i \oplus X)$, $Y^*_i = MAC_{F_i}(ID_i||SN'_j||T'_3)$ and checks the condition $Y^*_i = Y'_i$. If it holds, SN'_j responds to U_i's query (RM'), computes the session key $V'_i = h(ID_i||F_i||T'_5)$, $C'_i = h(RM')$ and $L' = E_{V'_i}(RM', C'_i)$, where T'_5 is the current timestamp of SN'_j, and finally sends the message $\langle L', T'_5\rangle$ to U_i via a public channel. Note that in this case, $\mathcal{A}$ can also derive the session key V'_i using X, ID_i, and T'_5. As a result, Althobaiti *et al.* scheme does not prevent the impersonation attacks.

4.2.3. Man-in-the-middle attack. In this attack, an adversary $\mathcal{A}$ tries to modify, delete, or change the contents of the messages so that the login-sensor nodes and U_i cannot detect them. Assume that $\mathcal{A}$ captures a login-sensor node and then he/she knows the secret parameter X from its memory. Suppose the GWN sends the message $\langle ID_i, Y_i, T_3\rangle$ to a login-sensor node SN_j from which U_i wants to obtain the response of the query. $\mathcal{A}$ intercepts this message, computes $F^*_i = h(ID_i \oplus X)$ using ID_i and extracted X, $Y^*_i = MAC_{F^*_i}(ID_i||SN_j||T^*_3)$, where T^*_3 is the current timestamp of $\mathcal{A}$'s system, and sends the modified message $\langle ID_i, Y^*_i, T^*_3\rangle$ to SN_j instead of the original message $\langle ID_i, Y_i, T_3\rangle$ via a public channel.

Upon reception of the message from $\mathcal{A}$, SN_j believes that the message is from the GWN and proceeds to validate the timestamp T^*_3, and if it is valid, SN_j computes $F_i = h(ID_i \oplus X)$, $Y^{**}_i = MAC_{F_i}(ID_i||SN_j||T^*_3)$ and checks the condition $Y^{**}_i = Y^*_i$. If it holds, SN_j responds to U_i's query

DOI: 10.1002/dac

(RM^*) by computing the session key shared with U_i as $V_i^* = h(ID_i||F_i||T_5^*)$, $C_i^* = h(RM^*)$ and $L^* = E_{V_i^*}(RM^*, and C_i^*)$ and sends the message $\langle L^*, T_5^* \rangle$, where T_5^* is the current timestamp of U_i's device. $\mathcal{A}$ again intercepts the message $\langle L^*, T_5^* \rangle$. $\mathcal{A}$ computes $V_i^{**} = h(ID_i||F_i^*||T_5^*)$ and decrypts L^* to retrieve RM^* and C_i^*. Note that $\mathcal{A}$ now knows the response to the query, RM^*, which is intended for U_i only. However, $\mathcal{A}$ can create a totally fake response RM^{**} to the query instead of the original RM^* and compute $C_i^{**} = h(RM^{**})$ and $L^{**} = E_{V_i^{**}}(RM^{**}, C_i^{**})$. Finally, $\mathcal{A}$ can send the modified message $\langle L^{**}, T_5^* \rangle$ to U_i. It is noted that this message is successfully authenticated by U_i, and hence, U_i treats RM^{**} as a valid response to his/her query. Thus, Althobaiti *et al.* scheme fails to protect this attack.

5. THE PROPOSED SCHEME

In this section, we aim to present a novel biometric-based user authentication scheme using fuzzy extractor method for biometric verification in order to withstand the security flaws found in Althobaiti *et al.* scheme. Our scheme has six phases: (i) pre-deployment; (ii) user registration; (iii) login; (iv) authentication and key agreement; (v) password and biometric update; and (vi) dynamic sensor node addition, which are described in the following detail. We use the notations listed in Table I.

5.1. Pre-deployment phase

Before the sensors are deployed in a particular target field, the GWN executes the following steps in offline.

Step PD1. The GWN selects a unique identity ID_{SN_j} for every deployed sensor SN_j.
Step PD2. The GWN then randomly chooses a master key MK_{SN_j} uniquely and computes a secret key $K_j = h(ID_{SN_j} \oplus MK_{SN_j})$ for each SN_j.
Step PD3. Finally, the GWN loads the information $\{ID_{SN_j}, K_j\}$ in the memory of SN_j.

At the end of this phase, the GWN stores the pair (ID_{SN_j}, K_j) in its database and then deletes MK_{SN_j}.

Remark 1
Note that the GWN knows the identities and master keys of all deployed sensor nodes. As in the threat model (provided in Section 4.1), the GWN is considered as trustworthy, and hence, we assume that the GWN will not reveal any keys to any adversary.

5.2. User registration phase

A legal user U_i needs to register to the GWN for purpose of accessing real-time data from login-sensor nodes later. It contains the following steps:

Step R1. U_i selects a unique identity ID_i, chooses a password PW_i, and then imprints his/her personal biometric B_i into a smart card reader.
Step R2. U_i chooses a random 1024-bit number K and a symmetric-key ek_i shared with the GWN. Note that K is only known to U_i.
Step R3. U_i computes the pseudo-random password $RPW_i = h(ID_i||PW_i||K)$ and sends the registration message $\langle ID_i, RPW_i, ek_i \rangle$ to the GWN via a secure channel.
Step R4. After receiving the registration request, the GWN selects a random 1024-bit secret key X_s, which is only known to it and then computes $f_i = h(ID_i \oplus h(X_s))$. The GWN then issues a smart card SC_i for U_i containing the information $\{f_i, h(\cdot), Gen(\cdot), Rep(\cdot), \mathcal{T}\}$ and sends it securely to U_i.
Step R5. Upon receiving the smart card SC_i from the GWN, U_i computes $Gen(B_i) = (\sigma_i, \tau_i)$, $f_i^* = f_i \oplus h(\sigma_i||ID_i||K) = h(ID_i \oplus h(X_s)) \oplus h(\sigma_i||ID_i||K)$, $e_i = h(ID_i||RPW_i||\sigma_i)$, $r_i = h(ID_i||\sigma_i) \oplus K$ and $BE_i = h(ID_i||\sigma_i) \oplus ek_i$. U_i then replaces f_i with f_i^* in

DOI: 10.1002/dac

Table III. User registration phase.

User (U_i)/Smart Card (SC_i)	GWN
Inputs ID_i, PW_i and imprints B_i. Generates a random 1024-bit secret number K. Computes $RPW_i = h(ID_i \|\| K \|\| PW_i)$. Chooses a symmetric-key ek_i. $\langle ID_i, RPW_i, ek_i \rangle$ $\xrightarrow{\quad\quad\quad\quad}$ (via a secure channel)	Generates a random 1024-bit secret key X_s. Computes $f_i = h(ID_i \oplus h(X_s))$. $\langle Smart\ Card(h(\cdot), Gen(\cdot), Rep(\cdot), f_i, \mathcal{T}) \rangle$ $\xleftarrow{\quad\quad\quad\quad}$ (via a secure channel)
Computes $Gen(B_i) = (\sigma_i, \tau_i)$, $f_i^* = f_i \oplus h(ID_i \|\| \sigma_i \|\| K)$, $e_i = h(ID_i \|\| RPW_i \|\| \sigma_i)$, $r_i = h(ID_i \|\| \sigma_i) \oplus K$, and $BE_i = h(ID_i \|\| \sigma_i) \oplus ek_i$. Replaces f_i with f_i^* in smart card. Stores $\{\tau_i, e_i, r_i, BE_i\}$ in smart card.	Stores ek_i corresponding to ID_i.

Table IV. Login phase.

User (U_i)/Smart Card (SC_i)	GWN
Inserts SC_i and inputs ID_i, PW_i and imprints B_i. Computes $\sigma_i' = Rep(B_i, \tau_i)$, $K' = r_i \oplus h(ID_i \|\| \sigma_i')$, $RPW_i' = h(ID_i \|\| PW_i \|\| K')$, and $e_i' = h(ID_i \|\| RPW_i' \|\| \sigma_i')$. Verifies if $e_i' = e_i$? If it holds, both password and biometric verification pass. $\langle ID_i, req \rangle$ $\xrightarrow{\quad\quad\quad\quad}$ (via a public channel)	

the smart card SC_i and also stores other information $\{\tau_i, e_i, r_i, BE_i\}$ in SC_i. Thus, SC_i finally containing the information $\{\tau_i, e_i, r_i, BE_i, f_i^*, h(\cdot), Gen(\cdot), Rep(\cdot), \mathcal{T}\}$.

At the end of this phase, the GWN inserts ek_i in its database corresponding to ID_i. The user registration phase is summarized in Table III.

5.3. Login phase

This phase involves the following steps by a legal user U_i in order to login to the GWN:

- Step L1. U_i first enters his/her smart card SC_i into a smart card reader of a specific terminal and imprints the biometric B_i. U_i then inputs his/her identity ID_i and password PW_i.
- Step L2. SC_i computes $\sigma_i' = Rep(B_i, \tau_i)$ using τ_i stored in the smart card and the fuzzy extractor function $Rep(\cdot)$, $K' = r_i \oplus h(ID_i \|\| \sigma_i')$, $RPW_i' = h(ID_i \|\| PW_i \|\| K')$ and $e_i' = h(ID_i \|\| RPW_i' \|\| \sigma_i')$. SC_i then verifies the condition $e_i' = e_i$. If it holds, SC_i ensures that both PW_i and B_i are correct, and hence, both password and biometric verification pass. Otherwise, this phase is aborted.
- Step L3. SC_i sends the login message $\langle ID_i, req \rangle$ to the GWN via a public channel, where *req* is a request.

The login phase is summarized in Table IV.

5.4. Authentication and key agreement phase

The purpose of this phase is to mutually authenticate each other and establish a secret session key for future secure communication. It has the following steps:

DOI: 10.1002/dac

Step AK1. When the login message $\langle ID_i, req\rangle$ is received by the GWN, it checks ID_i. If it is valid, the GWN generates a random challenge R and sends the authentication message $\langle R\rangle$ to U_i via a public channel.

Step AK2. When U_i receives the message in Step AK1, SC_i computes $ek_i = BE_i \oplus h(ID_i||\sigma_i')$, selects a login-sensor node, say SN_j from which U_i wants to access the real-time data, and sends the message $\langle E_{ek_i}(R, T_1, ID_{SN_j})\rangle$ to the GWN via a public channel, where T_1 is the current timestamp of U_i's device.

Step AK3. Upon receiving the message in Step AK2, the GWN decrypts $E_{ek_i}(R, T_1, ID_{SN_j})$ using the key ek_i stored in its database in order to retrieve R, T_1, and ID_{SN_j}. The GWN then verifies the validity of T_1 by the condition $|T_1 - T_1^*| < \Delta T$, where T_1^* is the time when the message is received by the GWN. If it is valid, the GWN further checks if the decrypted R matches with its previously generated challenge. If there is a match, the GWN continues. Otherwise, the GWN terminates the session.

Step AK4. The GWN computes $f_i^* = h(ID_i \oplus h(X_s))$, $f_i^{**} = h(ID_{SN_j} || f_i^*)$ and $Y_j = E_{K_j}[ID_i, ID_{SN_j}, T_1, T_2, f_i^{**}]$, where T_2 is the GWN's current timestamp. The GWN then sends the message $\langle ID_i, Y_j\rangle$ to the login-sensor node SN_j via a public channel.

Step AK5. Upon reception of the message $\langle ID_i, Y_j\rangle$, SN_j decrypts Y_j using its secret key K_j to retrieve ID_i, ID_{SN_j}, T_1, T_2, and f_i^{**} as $(ID_i^{''}, ID_{SN_j}^{''}, T_1^{''}, T_2^{''}, f_i^{''}) = D_{K_j}[Y_j]$. SN_j then checks the validity of the timestamp T_2 by the condition $|T_2 - T_2^*| < \Delta T$, where T_2^* is the time when the message is received by SN_j. If it is valid, SN_j further checks if $ID_i^{''}$ and $ID_{SN_j}^{''}$ match with ID_i in the received message and its own identity ID_{SN_j}, respectively, If all these conditions hold, SN_j computes a session key with U_i as $SK_{ij} = h(f_i^{''}||ID_i||ID_{SN_j}||T_1^{''}||T_3)$, where T_3 is the current timestamp of SN_j. After that, SN_j sends the message $\langle h(SK_{ij}), T_3\rangle$ back to U_i via a public channel.

Step AK6. Finally, after getting the message $\langle h(SK_{ij}), T_3\rangle$ from SN_j, SC_i of U_i checks the validity of the timestamp T_3 by the condition $|T_3 - T_3^*| < \Delta T$, where T_3^* is U_i's reception time of the message. SC_i then computes $f_i' = f_i^* \oplus h(\sigma_i'||ID_i||K')$, $f_i^{''} = h(ID_{SN_j}||f_i')$, and $SK_{ij}' = h(f_i^{''}||ID_i||ID_{SN_j}||T_1||T_3)$ and checks the condition $h(SK_{ij}') = h(SK_{ij})$. If this condition holds, SC_i confirms that SK_{ij} is the correct session key shared with the accessed login-sensor node SN_j, and hence, $SK_{ij}' = SK_{ij}$.

Finally, at the end of this phase, U_i and SN_j use SK_{ij}' and SK_{ij}, respectively, for their future secure communications. This phase is summarized in Table V.

5.5. *Password and biometric update phase*

A legal user U_i can change his/her old password PW_i and personal biometrics B_i as follows:

(1) Step PB1. U_i first inserts his/her smart card SC_i into a card reader of a specific terminal and then gives identity ID_i, old password PW_i and also imprints old biometrics B_i.

(2) Step PB2. SC_i then computes $\sigma_i' = Rep(B_i, \tau_i)$, $K' = r_i \oplus h(ID_i||\sigma_i')$, $RPW_i' = h(ID_i||PW_i||K')$, $e_i' = h(ID_i||RPW_i'||\sigma_i')$ and compares e_i' with e_i. If it holds, it ensures that both PW_i and B_i are correct. Otherwise, this phase is terminated.

(3) Step PB3. SC_i asks U_i to input new password PW_i^{new} and imprint new biometrics B_i^{new}. SC_i then generates a random 1024-bit number K^{new} and computes $RPW_i^{new} = h(ID_i||PW_i^{new}||K^{new})$, $Gen(B_i^{new}) = (\sigma_i^{new}, \tau_i^{new})$, $e_i^{new} = h(ID_i||RPW_i^{new}||\sigma_i^{new})$, $r_i^{new} = h(ID_i||\sigma_i^{new}) \oplus K^{new}$, $BE_i^{new} = (BE_i \oplus h(ID_i||\sigma_i')) \oplus h(ID_i||\sigma_i^{new}) = h(ID_i||\sigma_i^{new}) \oplus ek_i$, $f_i^{new} = (f_i^* \oplus h(\sigma_i'||ID_i||K')) \oplus h(\sigma_i^{new}||ID_i||K^{new}) = f_i \oplus h(\sigma_i^{new}||ID_i||K^{new}) = h(ID_i \oplus h(X_s)) \oplus h(\sigma_i^{new}||ID_i||K^{new})$.

(4) Step PB4. Finally, SC_i replaces τ_i, e_i, r_i, BE_i, and f_i^* with $\tau_i^{new}, e_i^{new}, r_i^{new}, BE_i^{new}$, and f_i^{new}, respectively.

DOI: 10.1002/dac

Table V. Authentication and key agreement phase.

User (U_i)/Smart Card (SC_i)	GWN	Sensor node (SN_j)
	Checks the validity of ID_i. If it passes, generates a random challenge R. $\langle R \rangle$ (via a public channel) $\longleftarrow$	
Computes $ek_i = BE_i \oplus h(ID_i \|\| \sigma_i')$. Selects a login-sensor node SN_j. $\langle E_{ek_i}(R, T_1, ID_{SN_j}) \rangle$ (via a public channel) $\longrightarrow$		
	Decrypts $E_{ek_i}(R, T_1, ID_{SN_j})$ using ek_i. Checks validity of T_1. If it is valid, checks decrypted R. Computes $f_i^* = h(ID_i \oplus h(X_s))$, $f_i^{**} = h(ID_{SN_j} \|\| f_i^*)$ and $Y_j = E_{K_j}[ID_i, ID_{SN_j}, T_1, T_2, f_i^{**}]$. $\langle ID_i, Y_j \rangle$ (via a public channel) $\longrightarrow$	
		Decrypts Y_j using secret key K_j. Checks validity of T_2, ID_i, ID_{SN_j}. If these are correct, computes session key $SK_{ij} = h(f_i'' \|\| ID_i \|\| ID_{SN_j} \|\| T_1'' \|\| T_3)$. $\langle h(SK_{ij}), T_3 \rangle$ (to U_i via a public channel) $\longleftarrow$
Checks validity of T_3. If it is valid, computes $f_i' = f_i^* \oplus h(\sigma_i' \|\| ID_i \|\| K')$, $f_i'' = h(ID_{SN_j} \|\| f_i')$ and $SK_{ij}' = h(f_i'' \|\| ID_i \|\| ID_{SN_j} \|\| T_1 \|\| T_3)$. Checks if $h(SK_{ij}') = h(SK_{ij})$? If so, stores SK_{ij}'.		Stores SK_{ij}.

5.6. Dynamic sensor node addition phase

Some sensor nodes may expire because of their battery consumption or they can be physically captured by an adversary. Suppose a new sensor node SN_j^{new} be deployed in the existing WSN. The following steps are involved during this phase:

- Step DA1. The GWN needs to select a unique identity $ID_{SN_j}^{new}$ for SN_j^{new}.
- Step DA2. The GWN then randomly chooses a master key $MK_{SN_j}^{new}$ uniquely and computes a secret key $K_j^{new} = h(ID_{SN_j}^{new} \oplus MK_{SN_j}^{new})$ for SN_j^{new}.
- Step DA3. After that, the GWN needs to pre-load $\{ID_{SN_j}^{new}, K_j^{new}\}$ in the memory of SN_j^{new} prior to its deployment.

Note that at the end of this phase, the GWN needs to store the pair $(ID_{SN_j}^{new}, K_j^{new})$ in its database and then delete $MK_{SN_j}^{new}$. Further, we do not need to update any information in SC_i for the added sensor node SN_j^{new}. The GWN then informs U_i about the addition of SN_j^{new} with the new identity $ID_{SN_j}^{new}$.

6. SECURITY ANALYSIS

This section analyzes both the informal and formal security analyses of our scheme.

6.1. Informal security analysis

Our scheme can resist various known attacks as described in the succeeding text.

DOI: 10.1002/dac

6.1.1. Stolen smart card attack. Suppose a legal user U_i has lost his/her smart card SC_i or it is stolen by an adversary $\mathcal{A}$. Then $\mathcal{A}$ can extract all the sensitive information $\{e_i, r_i, f_i^*, BE_i\}$ from SC_i using the power analysis attacks [43, 44], where $e_i = h(ID_i||RPW_i||\sigma_i)$, $RPW_i = h(ID_i||K||PW_i)$, $f_i^* = h(ID_i \oplus h(X_s)) \oplus h(\sigma_i||ID_i||K)$, $r_i = h(ID_i||\sigma_i) \oplus K$, and $BE_i = h(ID_i||\sigma_i) \oplus ek_i$. Note that K is 124-bit number kept secret to U_i only. Without knowing K, it is computationally infeasible to compute/derive PW_i due to difficulty of inverting $h(\cdot)$. Further, without knowing σ_i, it is computationally infeasible to know K. Then deriving X_s is also difficult for $\mathcal{A}$ without knowing K and σ_i. Hence, our scheme protects the stolen smart card attacks.

6.1.2. Replay attack. Assume that an adversary $\mathcal{A}$ eavesdrops the message $\langle E_{ek_i}(R, T_1, ID_{SN_j})\rangle$ during the authentication and key agreement phase and replays this message again after some time to the GWN. However, this message will be detected as a replay one due to validation of the timestamp T_1 by the GWN. Similarly, even if $\mathcal{A}$ eavesdrops the messages $\langle ID_i, Y_j\rangle$ and $\langle h(SK_{ij}), T_3\rangle$ during the authentication and key agreement phase and replays these messages to SN_j and U_i, respectively, after some time, they will be also detected as replay messages due to validation of the timestamps T_2 and T_3 at SN_j and U_i, respectively. Thus, our scheme protects the replay attacks due to usage of the timestamps in design.

6.1.3. Man-in-the-middle attack. Suppose an adversary $\mathcal{A}$ intercepts the message $\langle E_{ek_i}(R, T_1, ID_{SN_j})\rangle$ during the authentication and key agreement phase. Assume that $\mathcal{A}$ generates a new random challenge R^* and generates its own current timestamp T_1^*. However, $\mathcal{A}$ does not know the identity ID_{SN_j} of SN_j and also the encryption key ek_i. Without knowing both ID_{SN_j} and ek_i, it is a hard problem for $\mathcal{A}$ to modify the message $\langle E_{ek_i}(R, T_1, ID_{SN_j})\rangle$ as $\langle E_{ek_i}(R^*, T_1^*, ID_{SN_j})\rangle$. Thus, $\mathcal{A}$ does not have any ability to modify this message. Similarly, it is also computationally infeasible tasks for $\mathcal{A}$ to modify the messages $\langle ID_i, Y_j\rangle$ and $\langle h(SK_{ij}), T_3\rangle$, because $\mathcal{A}$ does not know the secret key K_j, $f_i^{''}$, ID_{SN_j}, and T_1. Hence, it is clear that our scheme protects the man-in-the-middle attack.

6.1.4. Many logged-in users with the same login-id attack. Assume that two users U_i and U_j choose the same password PW. Then the hash values $e_i = h(ID_i||RPW_i||\sigma_i)$ and $e_j = h(ID_j||RPW_j||\sigma_j)$ are different because of the properties of personal biometrics, random numbers selected by U_i and U_j, respectively, and ID_i and ID_j. In order to login to the GWN, U_i must have a valid tuple (ID_i, PW_i, B_i) and a smart card SC_i corresponding to these information. Thus, our scheme has the ability to prevent this attack.

6.1.5. Denial-of-service attack. In our scheme, during the authentication and key agreement phase, the GWN sends the message $\langle ID_i, Y_j\rangle$ to a login-sensor node SN_j from which U_i wants to access data. As a response, SN_j further sends an acknowledgment $\langle h(SK_{ij}), T_3\rangle$ to U_i for mutual authentication and also to confirm that U_i successfully establishes the same secret session key with SN_j. If an adversary $\mathcal{A}$ blocks the messages from reaching the GWN and the sensors, they will know about malicious dropping of these messages. Furthermore, if $\mathcal{A}$ does the malicious flooding of authentication requests to the sensors, the corresponding sensor nodes need to perform at most one symmetric-key decryption and two hashings only. However, such an attack can be also detected by enabling the network with an intrusion–detection system. In addition, the efficiency of Advanced Encryption Standard (AES) symmetric-key decryption and hashing allows the authentication of each message not to consume too much energy, space, and computation resources. Hence, our scheme can resist this attack.

6.1.6. Offline password guessing attack. Assume that the smart card SC_i of a legal user U_i is lost or stolen by an adversary $\mathcal{A}$. Then $\mathcal{A}$ can know the sensitive information $\{e_i, r_i, f_i^*, BE_i\}$ from SC_i using the power analysis attacks [43, 44]. Note that $e_i = h(ID_i||RPW_i||\sigma_i)$ and $RPW_i = h(ID_i||K||PW_i)$, where K is a 1024-bit secret number of U_i only. Without knowing K, it is difficult for $\mathcal{A}$ to guess correctly the password PW_i of U_i due to application of $h(\cdot)$. Thus, our scheme protects this attack.

DOI: 10.1002/dac

6.1.7. Online password guessing attack. Let an adversary $\mathcal{A}$ intercept the message $\langle ID_i, req\rangle$ during the login phase and other messages $\langle R\rangle$, $\langle E_{ek_i}(R, T_1, ID_{SN_j})\rangle$, $\langle ID_i, Y_j\rangle$, and $\langle h(SK_{ij}), T_3\rangle$ during the authentication and key agreement phase. Note that none of the messages involve the password PW_i of a legal user U_i directly or indirectly. Thus, there is no way for $\mathcal{A}$ to derive PW_i of U_i through this attack.

6.1.8. Privileged-insider attack. In our scheme, during the user registration phase, a legal user U_i sends the registration request message $\langle ID_i, RPW_i, ek_i\rangle$ to the GWN securely, where $RPW_i = h(ID_i \,||PW_i\, ||K)$ and K is a 1024-bit random secret number chosen by U_i only. Without knowing K, it is a difficult task to guess the password PW_i of U_i correctly because of difficulty of inverting the collision-resistant hash function $h(\cdot)$. It is thus clear that our scheme protects the privileged-insider attack from deriving the password PW_i of U_i.

6.1.9. Impersonation attack. Suppose an adversary $\mathcal{A}$ physically captures a login-sensor node, say SN_j. $\mathcal{A}$ then knows the secret key K_j and the identity ID_{SN_j} from the captured node SN_j, where $K_j = h(ID_{SN_j} \oplus MK_{SN_j})$. Note that the master key MK_{SN_j} generated by the GWN for each sensor node SN_j is distinct, and thus, each K_j is also distinct. Further, assume that $\mathcal{A}$ intercepts the message $\langle ID_i, Y_j\rangle$ during our authentication and key agreement phase. Suppose $\mathcal{A}$ wants to impersonate the GWN to another non-compromised login-sensor node, say SN'_j whose secret key is $K'_j = h(ID_{SN'_j} \oplus MK'_{SN_j})$, where ID'_{SN_j} and MK'_{SN_j} are the identity and master key for SN'_j. Then he/she does not have the ability to reconstruct the message $\langle ID_i, Y'_j\rangle$ for SN'_j as $f_i^{**} = h(ID_{SN'_j} \,||f_i^*)$ and $Y'_j = E_{K'_j}[ID_i, ID'_{SN_j}, T'_1, T'_2, f_i^{**}]$, where T_1 and T_2 are the timestamps generated by $\mathcal{A}$. Our scheme thus protects this attack.

6.1.10. Resilience against node capture attack. Assume that an adversary $\mathcal{A}$ physically captures a sensor node, say SN_j randomly or selectively from a target field in the network from which a legal user U_i wishes to access data. Note that the sensors are not tamper-resistant, and thus, $\mathcal{A}$ can easily compromise all the secret information including the captured sensor node's secret key K_j, identity ID_{SN_j}, and also session key shared with U_i. The secret session key generated between SN_j and U_i is computed by $f_i'' = h(ID_{SN_j}||h(ID_i \oplus h(X_s)))$, T_1, T_3, and ID_i, because each established session key between a user and a sensor node is unique because of usage of timestamps in our design. Note that each SN_j is given prior to its deployment a uniquely random secret key K_j. $\mathcal{A}$ is then able to compromise K_j of that captured SN_j only. As a result, $\mathcal{A}$ can contact with U_i only with bogus data. Note that other uncaptured sensor nodes can still contact securely with the actual real-time readings to their corresponding authenticated users. This means that capture of a sensor node does not leak any other secret information about other uncaptured sensor nodes and users so that $\mathcal{A}$ can compromise secure communication between the users and those uncaptured nodes. This property is known as unconditional security against node capture attack.

6.2. Formal security analysis

In this section, we evaluate our scheme for the formal security analysis against an adversary in the generic group model. We follow the formal security analysis similar to that presented in [34, 37]. For this purpose, we use the method of contradiction proof [46] in our analysis. One can also prove the formal security in the standard model. However, in this paper, we have performed the formal security analysis under the generic group model of cryptography.

In order to apply the method of contradiction, we assume that there exist the following two random oracles for an adversary:

- *Reveal1*. This oracle will unconditionally output the input string x from the corresponding hash digest value $y = h(x)$.
- *Reveal2*. This oracle will unconditionally output the plaintext message m using a symmetric-key cryptosystem Ω without knowing the key k, which uses only the public parameters and ciphertext message $E_k(m)$.

DOI: 10.1002/dac

Theorem 1
If a one-way hash function $h(\cdot)$ closely behaves like a random oracle, our scheme is secure against an adversary for deriving the encryption key ek_i shared between a legal user U_i and the GWN, the password PW_i of U_i, and the secret key X_s of the GWN, even if the smart card SC_i of U_i is lost or stolen.

Proof
We first construct an adversary $\mathcal{A}$ who will have the ability to derive the encryption key ek_i shared between a legal user U_i and the GWN, the password PW_i of U_i, and the secret key X_s of the GWN. Suppose the smart card SC_i of U_i is lost/stolen. Then $\mathcal{A}$ can know the information from SC_i by performing the power analysis attacks [43, 44]. $\mathcal{A}$ can run the experiment called $EXP1^{HASH}_{BAS,\mathcal{A}}$ for our biometric authentication scheme (BAS), which is provided in Algorithm 1. Note that this experiment uses the *Reveal1* oracle to derive ek_i, PW_i, and X_s. We define the success probability for $EXP1^{HASH}_{BAS,\mathcal{A}}$ as $Succ1^{HASH}_{BAS,\mathcal{A}} = |Pr[EXP1^{HASH}_{BAS,\mathcal{A}} = 1] - 1|$. The advantage function for $EXP1^{HASH}_{BAS,\mathcal{A}}$ is $Adv1(t_1, q_{R_1}) = \max_{\mathcal{A}}\{Succ1^{HASH}_{BAS,\mathcal{A}}\}$, where the maximum is considered over all $\mathcal{A}$ with execution time t_1, and q_{R_1} is the total queries made to the *Reveal1* oracle. The proposed scheme is secure against $\mathcal{A}$ for deriving ek_i, PW_i, and X_s, if $Adv1(t_1, q_{R_1}) \leqslant \epsilon_1$, for any sufficiently small value of ϵ_1.

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

1: Extract the information $\{e_i, r_i, f_i^*, BE_i\}$ from the lost/stolen smart card SC_i of U_i using the power analysis attacks [43, 44], where $e_i = h(ID_i||RPW_i||\sigma_i)$, $RPW_i = h(ID_i||K||PW_i)$, $f_i^* = h(ID_i \oplus h(X_s)) \oplus h(\sigma_i||ID_i||K)$, $r_i = h(ID_i||\sigma_i) \oplus K$, and $BE_i = h(ID_i||\sigma_i) \oplus ek_i$.
2: Use *Reveal1* oracle on input e_i to get ID_i, RPW_i and σ_i as $(ID_i'||RPW_i'||\sigma_i') \leftarrow Reveal1(e_i)$.
3: Eavesdrop the login message $\langle ID_i, req\rangle$ during the login phase.
4: **if** $(ID_i' = ID_i)$ **then**
5: Use σ_i' to compute the encryption key ek_i as $ek_i' = BE_i \oplus h(ID_i'||\sigma_i')$, and the secret number K as $K' = r_i \oplus h(ID_i'||\sigma_i')$.
6: Compute $u = f_i^* \oplus h(\sigma_i'||ID_i'||K')$.
7: Call *Reveal1* oracle on input u to compute $v \leftarrow Reveal1(v)$, which needs to be $ID_i \oplus h(X_s)$.
8: Use ID_i' to compute $w = v \oplus ID_i'$.
9: Retrieve X_s as $X_s' \leftarrow Reveal1(w)$ using the *Reveal1* oracle on input w.
10: Eavesdrop the messages $\langle R\rangle$ and $\langle E_{ek_i}(R, T_1, ID_{SN_j})\rangle$ during the authentication and key agreement phase.
11: Decrypt $E_{ek_i}(R, T_1, ID_{SN_j}$ to derive R using the computed key ek_i' in Step 5. Let it be R^*.
12: Call *Reveal1* oracle on input RPW_i' in Step 2 to retrieve ID_i, K and PW_i as $(ID_i^*||K^*||PW_i^*) \leftarrow Reveal1(RPW_i')$.
13: **if** $(K^* = K')$ and $(R^* = R)$ **then**
14: Accept ek_i', PW_i^* and X_s' as the correct encryption key ek_i, password PW_i of U_i and secret key X_s of the GWN, respectively.
15: **return** 1
16: **else**
17: **return** 0
18: **end if**
19: **else**
20: **return** 0
21: **end if**

Considering the experiment given in Algorithm 1, we see that if there exists the *Reveal1* oracle that can invert $h(\cdot)$, $\mathcal{A}$ can easily derive ek_i, PW_i, and X_s. However, according to Definition 1, it is a difficult job for $\mathcal{A}$ to invert $h(\cdot)$, that is, $Adv^{HASH}_{\mathcal{A}}(t) \leqslant \tau$, for any sufficiently small $\tau > 0$. Thus, it is a contradiction. This contradiction shows that we cannot have such an adversary $\mathcal{A}$, who will have the ability to invert $h(\cdot)$ and also to derive ek_i, PW_i, and X_s. As a result, we have $Adv1(t_1,$

DOI: 10.1002/dac

$q_{R_1}) \leqslant \epsilon_1$, for any sufficiently small value of ϵ_1, because it is dependent on $Adv_{\mathcal{A}}^{HASH}(t)$. Hence, the theorem is proved. □

Theorem 2
If the used symmetric-key encryption scheme, Ω, is IND-CPA secure, our scheme is secure against an adversary for deriving the session key SK_{ij} between a legal user U_i and a login-sensor node SN_j.

Proof
The proof of this theorem is similar to that presented in Theorem 1. We construct an adversary $\mathcal{A}$ who will have the ability to derive the session key SK_{ij} between a legal user U_i and a login-sensor node SN_j. For this purpose, $\mathcal{A}$ can run the experiment $EXP2_{BAS,\mathcal{A}}^{IND\text{-}CPA}$ for our biometric authentication scheme (BAS), which is given in Algorithm 2. This experiment uses the *Reveal2* oracle to derive SK_{ij}. We define the success probability for $EXP2_{BAS,\mathcal{A}}^{IND\text{-}CPA}$ as $Succ2_{BAS,\mathcal{A}}^{IND\text{-}CPA} = |Pr[EXP2_{BAS,\mathcal{A}}^{IND\text{-}CPA} = 1] - 1|$. The advantage function for $EXP2_{BAS,\mathcal{A}}^{IND\text{-}CPA}$ is $Adv2(t_2, q_{R_2}) = \max_{\mathcal{A}}\{Succ2_{BAS,\mathcal{A}}^{IND\text{-}CPA}\}$, where t_2 is the execution time and q_{R_2} the total queries made to the *Reveal2* oracle. The proposed scheme is secure against $\mathcal{A}$ for deriving SK_{ij}, if $Adv2(t_2, q_{R_2}) \leqslant \epsilon_2$, for any sufficiently small value of ϵ_2.

Algorithm 2 $EXP2_{BAS,\mathcal{A}}^{IND-CPA}$

1: Eavesdrop the message $\langle ID_i, Y_j \rangle$ during the authentication and key agreement phase.
2: Use *Reveal2* oracle on input Y_j to extract ID_i, ID_{SN_j}, T_1, T_2, and f_i^{**} as $(ID_i' || ID_{SN_j}' || T_1' || T_2' || f_i') \leftarrow Reveal2(Y_j)$.
3: **if** $(ID_i' = ID_i)$ **then**
4: Eavesdrop the message $\langle h(SK_{ij}), T_3 \rangle$ during the authentication and key agreement phase.
5: Compute $SK_{ij}^* = h(f_i' || ID_i' || ID_{SN_j}' || T_1' || T_3)$.
6: **if** $(h(SK_{ij}^*) = h(SK_{ij}))$ **then**
7: Accept SK_{ij}^* as the correct session key shared between U_i and SN_j.
8: **return** 1
9: **else**
10: **return** 0
11: **end if**
12: **else**
13: **return** 0
14: **end if**

From the experiment provided in Algorithm 2, note that if there exists the *Reveal2* oracle, $\mathcal{A}$ can easily derive SK_{ij}. However, according to Definition 2 (provided in Section 2.2), it is computationally infeasible problem, that is, the encryption scheme Ω is IND-CPA secure in the single (multiple) eavesdropper setting if $Adv_{\Omega,SE}^{IND\text{-}CPA}(l)$ (resp. $Adv_{\Omega,ME}^{IND\text{-}CPA}(l)$) is negligible (in the security parameter l). It is again a contradiction, which shows that we cannot have such $\mathcal{A}$ to derive SK_{ij}. Hence, we have $Adv2(t_2, q_{R_2}) \leqslant \epsilon_2$, for any sufficiently small value of ϵ_2, because it is again dependent on $Adv_{\Omega,SE}^{IND\text{-}CPA}(l)$. It then follows the proof. □

7. SIMULATION FOR FORMAL SECURITY VERIFICATION

In this section, we simulate our scheme for the formal security verification using the widely accepted and used AVISPA tool and then show that our scheme prevents the replay and man-in-the-middle attacks.

Automated Validation of Internet Security Protocols and Applications is used for the automated validation of Internet security-sensitive protocols and applications. It integrates four back ends that implement a variety of state-of-the-art automatic analysis techniques [47, 48], which are (i) OFMC; (ii) CL-AtSe; (iii) SATMC; and (iv) TA4SP. The detailed description and functionality of these back ends could be found in [47, 48]. To test a protocol whether it is secure or not, it needs to be implemented in High Level Protocols Specification Language (HLPSL) [49]. This high level

DOI: 10.1002/dac

description is then converted to intermediate format (IF) using the HLPSL2IF translator. HLPSL is a role-oriented language. Each role is not dependent on the other roles. The intruder is modeled using the Dolev–Yao model [45] so that the intruder can play as an authorized player in a protocol run.

The output format is produced by applying one of the four back ends. The output provides precise information about the result and the conditions for which it is obtained. The analysis of a scheme is treated as successful if there is an attack or not. Various sections in output format are given as follows:

- SUMMARY. It means that whether a scheme is secure, insecure, or whether the security analysis is inconclusive (may or may not be safe).
- DETAILS. It explains the conditions whether the tested scheme is secure, insecure, or inconclusive.
- PROTOCOL, GOAL, and BACK END. These present the name of the scheme, goal of the analysis, and name of back end applied, respectively.
- In the standard format, the trace of an attack (if any) is also shown.

High Level Protocols Specification Language supports various basic types, and some of them are given in the succeeding text for the specifications of different roles provided in the next subsection.

- *agent*. It is a principal name. The special identifier i is always for intruder.
- *public_key*. It means the agents' public keys in a public-key cryptosystem. If ku is a public key, its private key is derived using *inv_ku*.
- *symmetric_key*. It is a key for a symmetric-key cryptosystem.
- *channel(dy)*. This data type is for the Dolev–Yao channel.
- *text*. It is used for nonces. These values are also used for messages. If Ra is of type *text (fresh)*. An intruder cannot guess a fresh value Ra'.
- *nat*. It represents the natural numbers in non-message contexts.
- *const*. It represents constants.
- *hash_func*. It represents a cryptographic one-way collision-resistant hash function.

The space of legal messages is defined as the closure of the basic types. For a given message m and encryption key k, $\{m\}\_k$ is the symmetric/public-key encryption. The associative '.' operator is always used for concatenation purpose. The declaration '*played_by X*' means that an agent X plays in a specific role. The top-level *Environment* role specifies the intruder's initial knowledge. Immediate reaction transitions are of the form $A = | > B$, which relate an event A and an action B. $A = | > B$ tells that if a transition is labeled in such a way as to make the event predicate A true, we must immediately (that is, simultaneously) perform action B. If a variable V needs to be permanently kept secret, the goal *secrecy_of V* needs to be used. One can refer to the detailed descriptions of AVISPA and HLPSL in [47, 48].

7.1. Specifying our scheme in HLPSL

We describe our implementation aspects in HLPSL language. We have three basic roles: alice for the user U_i, server for the GWN, and bob for the login-sensor node SN_j. Apart from these, we have another two roles, which are for the session and the environment. In Figure 2, U_i first receives the start signal and changes its state from 0 to 1 and then initiates the communication with the GWN by sending the registration request $\langle ID_i, RPW_i, ek_i \rangle$ to the GWN securely with the help of *Snd*( ) operation. In response, U_i obtains the smart card having the required information from the GWN securely using *Rcv*( ) operation. During the login phase, U_i sends the login request $\langle ID_i, req \rangle$ to the GWN. During our authentication and key agreement phase, U_i first obtains the challenge $\langle R \rangle$ from the GWN via a public channel, and as a response, U_i sends $\langle E_{ek_i}(R, T_1, ID_{SN_j}) \rangle$ to the GWN back via a public channel. After that, U_i receives $\langle h(SK_{ij}), T_3 \rangle$ from SN_j via a public channel. witness(Ui, GWN, alice_server_t1, T1') declares that U_i has freshly generated the current timestamp T_1 for the GWN, which is included in the message $\langle E_{ek_i}(R, T_1, ID_{SN_j}) \rangle$. request(SNj, Ui, bob_alice_t3, T3') tells U_i's acceptance of the timestamp T_3 generated for U_i by SN_j, which is included in the message $\langle h(SK_{ij}), T_3 \rangle$.

DOI: 10.1002/dac

```
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, T1, T2, T3: text,
     Xs, EKi, Kj, Request, R, RPWi : text,
     Gen, Rep : hash_func
  const alice_server_t1, server_bob_t2,
        bob_alice_t3, sub1, sub2, sub3, sub4 : protocol_id
init  State := 0
transition
1. State   = 0 /\ Rcv(start) =|>
% Registration phase
State' := 1    /\ K' := new()
              /\ secret({PWi,Bi,K'},sub1,Ui)
              /\ secret(EKi,sub2,{Ui,GWN})
              /\ RPWi' := H(IDi.PWi.K')
% Ui sends login message to GWN securely
              /\ Snd({IDi.RPWi'.EKi}_SKuigwn)
% Ui receives the smart card from GWN securely
2. State = 1 /\ Rcv({H.Gen.Rep.H(xor(IDi,H(Xs)))}_SKuigwn) =|>
% Login phase
State' := 2  /\ secret(Xs,sub3,GWN)
% Ui sends the login message to the GWN
             /\ Snd(IDi.Request)
% Authentication and key agreement phase
% Ui receives the message <R> from GWN
3. State = 2   /\ Rcv(R')  =|>
   State' := 3 /\ T1' := new()
% Ui sends the message <E_eki(R,T1,IDsnj)> to GWN
             /\ Snd({R'.T1'.IDsnj}_EKi)
% Ui has freshly generated the value T1 for GWN
             /\ witness(Ui,GWN,alice_server_t1, T1')
% Ui receives the message from sensor node SNj
2. State = 3  /\ Rcv(H(H(H(IDsnj.H(xor(IDi,H(Xs))))).
                 IDi.IDsnj.T1'.T3').T3')) =|>
% Ui's acceptance of the value T3 generated for Ui by SNj
State' := 4 /\ request(SNj, Ui, bob_alice_t3, T3')
end role
```

Figure 2. Role specification for the user U_i.

In Figure 3, we have shown the HLPSL implementation of our scheme for the role of the GWN. In this role, during the registration phase, after receiving the registration request $\langle ID_i, RPW_i, ek_i \rangle$ securely from U_i, the GWN sends the smart card containing the necessary information securely to U_i. After receiving the login message $\langle ID_i, req \rangle$ from U_i, the GWN sends the challenge $\langle R \rangle$ to U_i. As a response, the GWN receives the message $\langle E_{ek_i}(R, T_1, ID_{SN_j}) \rangle$ from U_i. The GWN then sends the message $\langle ID_i, Y_j \rangle$ to SN_j. In this role, the GWN accepts the value T_1 generated for GWN by U_i, which is included in the message $\langle E_{ek_i}(R, T_1, ID_{SN_j}) \rangle$. Note that secret(Xs, sub3, GWN) declaration

DOI: 10.1002/dac

```
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, T1, T2, T3: text,
    Xs, EKi, Kj, Request, R, RPWi : text,
    Gen, Rep : hash_func
  const alice_server_t1, server_bob_t2,
      bob_alice_t3, sub1, sub2, sub3, sub4 : protocol_id
init  State := 0
transition
% Registration phase
% GWN receives login message from UI securely
1. State = 0 /\ Rcv({IDi.H(IDi.PWi.K').EKi}_SKuigwn) =|>
   State' := 1 /\ secret({PWi,Bi,K'},sub1,Ui)
% GWN sends the smart card to Ui securely
           /\ Snd({H.Gen.Rep.H(xor(IDi,H(Xs)))}_SKuigwn)
% Login phase: receive the login request message from Ui
2. State = 1 /\ Rcv(IDi.Request) =|>
State' := 2 /\ R' := new( )
          /\ secret(EKi,sub2,{Ui,GWN})
          /\ secret(Xs,sub3,GWN)
          /\ secret(Kj,sub4,{GWN,SNj})
% Authentication and key agreement phase
% GWN sends the message to Ui
          /\ Snd(R')
% GWN receives the message from Ui
3. State = 2 /\ Rcv({R'.T1'.IDsnj}_EKi) =|>
 State' := 3 /\ T2' := new()
% GWN sends the message to SNj
           /\ Snd(IDi.{IDi.IDsnj.T1'.T2'.
           H(IDsnj.H(xor(IDi,H(Xs))))}_Kj)
% GWN has freshly generated the value T2 for SNj
          /\ witness(GWN,SNj,server_bob_t2, T2')
% GWN's acceptance of the value T1 generated for GWN by Ui
          /\ request(Ui, GWN, alice_server_t1, T1')
end role
```

Figure 3. Role specification for the GWN.

means that the GWN keeps the secret information X_s to itself only, which is characterized by the protocol id sub3.

In Figure 4, we have specified the role for SN_j. In this role, SN_j receives $\langle ID_i, Y_j \rangle$ from the GWN during the authentication and key agreement phase. After that, SN_j sends $\langle h(SK_{ij}), T_3 \rangle$ to U_i. In this role, SN_j verifies the GWN based on T_2 included in the message $\langle ID_i, Y_j \rangle$ by the declaration request(GWN, SNj, server_bob_t2, T2').

In Figure 5, we have implemented the role specification in HLPSL for the session. Figure 6 shows the role specification for the goal and the environment. In the session segment, all the basic roles

DOI: 10.1002/dac

```
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, T1, T2, T3: text,
     Xs, EKi, Kj, Request, R, RPWi : text,
     Gen, Rep : hash_func
  const alice_server_t1, server_bob_t2,
        bob_alice_t3, sub1, sub2, sub3, sub4 : protocol_id
init  State := 0
transition
% Authentication and key agreement phase
% Receive the message from the GWN
1. State = 0 /\ Rcv(IDi.{IDi.IDsnj.T1'.T2'.
            H(IDsnj.H(xor(IDi,H(Xs))))}_Kj) =|>
State' := 1 /\ T3' := new()
          /\ secret({PWi,Bi,K},sub1,Ui)
          /\ secret(EKi,sub2,{Ui,GWN})
          /\ secret(Xs,sub3,GWN)
          /\ secret(Kj,sub4,{GWN,SNj})
% Send the message to Ui
          /\ Snd(H(H(H(IDsnj.H(xor(IDi,H(Xs))))).
              IDi.IDsnj.T1'.T3').T3'))
% SNj has freshly generated the value T3 for SNj
          /\ witness(SNj,Ui,bob_alice_t3, T3')
% SNj's acceptance of the value T2 generated for SNj by GWN
          /\ request(GWN, SNj, server_bob_t2, T2')
end role
```

Figure 4. Role specification for the login-sensor node SN_j.

```
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
```

Figure 5. Role specification for the session.

DOI: 10.1002/dac

```
role environment()
def=
 const ui, gwn, snj : agent,
       h, gen, rep : hash_func,
       skuigwn: symmetric_key,
       idi, idsnj, t1, t2, t3 : text,
       alice_server_t1, server_bob_t2,
       bob_alice_t3, sub1, sub2,
       sub3, sub4 : protocol_id
intruder_knowledge ={idi,h,gen,rep,t3}
  composition
session(ui, gwn, snj, h, skuigwn)
   /\session(ui, gwn, snj, h, skuigwn)
   /\  session(ui, gwn, snj, h, skuigwn)
end role

goal
  secrecy_of sub1
  secrecy_of sub2
  secrecy_of sub3
  secrecy_of sub4
  authentication_on alice_server_t1
  authentication_on server_bob_t2
  authentication_on bob_alice_t3
end goal
environment()
```

Figure 6. Role specification for the goal and the environment.

including the roles for U_i, GWN, and SN_j are treated as instances with concrete arguments. The top-level role (environment) includes the global constants and a composition of one or more sessions. The intruder can also play some roles as legitimate users. We have verified four secrecy goals and three authentications.

7.2. Simulation results

We have chosen the widely accepted OFMC back end [50] for the execution tests with a bounded number of sessions model checking. For the replay attack checking, OFMC verifies if a legitimate party can execute a specified scheme as a passive intruder. OFMC then provides the attacker the knowledge of some normal sessions between legitimate agents. For the Dolev–Yao model checking, OFMC checks if any man-in-the-middle attack is possible by the attacker. Finally, our scheme is simulated under OFMC back end using AVISPA web tool [51]. The formal security verification results provided in Figure 7 clearly ensure that the proposed scheme is secure against the active attacks such as replay and man-in-the-middle attacks.

8. PERFORMANCE AND FUNCTIONALITY COMPARISON

This section compares the security features and performance of our scheme with those of other related recently proposed schemes of Althobaiti *et al.* [29], Yoo *et al.* [52], Sun *et al.* [53], Xue *et al.* [54], and Jiang *et al.* [55].

DOI: 10.1002/dac

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/avispa/web−interface−computation/
./tempdir/workfileURTTEp.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 1.10s
  visitedNodes: 158 nodes
  depth: 6 plies
```

Figure 7. The result of the analysis using OFMC of our scheme.

Table VI. Features comparison.

Security features	[29]	[52]	[53]	[54]	[55]	Ours
SF_1	Yes	Yes	Yes	No	No	Yes
SF_2	Yes	Yes	Yes	Yes	Yes	Yes
SF_3	N/A	Yes	Yes	Yes	Yes	Yes
SF_4	Yes	No	Yes	No	Yes	Yes
SF_5	Yes	Yes	Yes	Yes	Yes	Yes
SF_6	Yes	Yes	Yes	Yes	Yes	Yes
SF_7	Yes	Yes	No	Yes	Yes	Yes
SF_8	No	Yes	Yes	Yes	Yes	Yes
SF_9	No	Yes	No	No	No	Yes
SF_{10}	No	No	No	No	No	Yes
SF_{11}	Yes	No	No	No	No	Yes
SF_{12}	No	Yes	Yes	Yes	Yes	Yes
SF_{13}	No	Yes	Yes	Yes	Yes	Yes
SF_{14}	No	Yes	Yes	Yes	Yes	Yes
SF_{15}	No	No	No	No	No	Yes
SF_{16}	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 provides mutual authentication; SF_8, whether supports key agreement between U_i and SN_j; SF_9, whether supports correct password update; SF_{10}, whether supports correct biometric update; SF_{11}, whether provides non-repudiation; SF_{12}, whether resilient against node capture attack; SF_{13}, whether resilient against impersonation attack; SF_{14}, whether secure against man-in-the-middle attack; SF_{15}, whether provides formal security analysis and verification; and SF_{16}, whether supports dynamic sensor node addition.

DOI: 10.1002/dac

Table VII. Computational overhead comparison.

Phase	Entity	[29]	[52]	[53]	[54]	[55]	Ours
UR	U_i	$t_{bfe} + 2t_h$	t_h	–	$2t_h$	t_h	$t_{fe} + 4t_h$
	GWN	t_h	$3t_h$	$2t_h$	$4t_h$	t_h	$2t_h$
L	U_i	$t_{bfe} + 4t_h + 2t_{enc}/t_{dec}$	$5t_h$	$2t_h$	$10t_h$	$7t_h$	$t_{fe} + t_{enc} + 6t_h$
+	GWN	$t_{enc} + t_{mac} + 3t_h$	$8t_h$	$5t_h$	$13t_h$	$10t_h$	$2t_{enc}/t_{dec} + 3t_h$
A	SN_j	$t_{dec} + t_{mac} + t_h$	$2t_h$	$2t_h$	$6t_h$	$5t_h$	$t_{dec} + 2t_h$

Note: UR, user registration phase; L, login phase; A, authentication phase.

Table VIII. Communication overhead comparison.

Scheme	Communication overhead
[29]	Five messages (864 bits)
[52]	Six messages (1824 bits)
[53]	Five messages (1296 bits)
[54]	Four messages (2256 bits)
[55]	Four messages (1920 bits)
Ours	Five messages (832 bits)

Table VI shows the security features provided by our scheme and other schemes. It is clear from this table that our scheme supports extra important security features as compared with other schemes [29, 52–55]. In our scheme, the password and biometric updates always take place correctly, whereas in other schemes, these features are not there. Althobaiti *et al.* scheme [29] and our scheme provide non-repudiation because of biometrics application. The proposed scheme provides the formal security verification using AVISPA tool, whereas other schemes do not provide it. Moreover, dynamic sensor node addition after initial deployment is supported in our scheme, which is considered as a very essential feature in designing security protocols in WSNs.

In Table VII, the computational cost comparison is shown for the user registration, login, and authentication phases. We denote t_{bfe}, t_h, t_{enc}, t_{dec}, t_{fe}, and t_{mac} as the time needed to perform biometric feature extraction, one-way hashing operation, symmetric-key encryption, symmetric-key decryption, fuzzy extractor operation ($Gen(\cdot)$ or $Rep(\cdot)$), and message authentication code (MAC) operation, respectively. Because of computational efficiency of the fuzzy extractor operations, our scheme is also comparable with related schemes. Note that in our scheme, only two hash functions $h(\cdot)$ and one symmetric-key decryption $D_k(\cdot)$ operation are needed for a sensor node SN_j during the authentication phase. Further, because of efficiency of $h(\cdot)$ and $D_k(\cdot)$, it is clear that our scheme is very much suitable for resource-constrained WSNs.

Finally, in Table VIII, we have shown communication cost of our scheme and other schemes during the login and authentication phases. We have assumed that ID_i, req, R, $E_k(\cdot)$, timestamp T_i, and $h(\cdot)$ require 160, 32, 32, 128 (if we use AES encryption), 32, and 160 bits, respectively. In our scheme, during the login phase, the message $\langle ID_i, req \rangle$ requires 192 bits, whereas during the authentication and key agreement phases, the messages $\langle R \rangle$, $\langle E_{ek_i}(R, T_1, ID_{SN_j}) \rangle$, $\langle ID_i, Y_j \rangle$, and $\langle h(ID_{ID_{ij}}, T_3) \rangle$ require 32, 128, 288, and 352 bits, respectively. As a result, the total communication cost of our scheme becomes 832 bits. On the other hand, the communication overheads of Althobaiti *et al.* scheme [29], Yoo *et al.* scheme [52], Sun *et al.* scheme [53], Xue *et al.* scheme [54], and Jiang *et al.* scheme [55] are 864, 1824, 1296, 2256 and 1920 bits, respectively. It is thus clear that our scheme is efficient.

9. CONCLUSION

We have first reviewed Althobaiti *et al.* scheme for WSNs. Unfortunately, we have shown that their scheme is insecure against several known attacks. We have then proposed a novel approach for the

DOI: 10.1002/dac

user authentication problem using the smart card. Our scheme withstands the security weaknesses found in Althobaiti *et al.* scheme. Through the rigorous security analysis theoretically and using simulation, we have shown that our scheme satisfies the desirable security requirements. Further, our scheme supports extra features as compared with Althobaiti *et al.* scheme and other related schemes. Correct password and biometric update and dynamic node addition phases are supported in our scheme, whereas other schemes do not support these important features. Overall, the security features, high security, and communication and computation efficiency of our scheme make our scheme very suitable for WSNs. In the future, we would like to design efficient and secure temporal credential-based three-factor user authentication approaches for WSNs, which could provide more security and functionality features as compared with those of other existing approaches.

ACKNOWLEDGEMENTS

The author is thankful to the reviewers, the editor, and the editor-in-chief for their valuable suggestions.

REFERENCES

1. Liu G, Xu B, Chen H. An indicator kriging method for distributed estimation in wireless sensor networks. *International Journal of Communication Systems* 2014; **27**(1):68–80.
2. Xue L, Guan X, Liu Z, Yang B. TREE: routing strategy with guarantee of QOS for industrial wireless sensor networks. *International Journal of Communication Systems* 2014; **27**(3):459–481.
3. Gao D, Zhu W, Xu X, Chao HC. A hybrid localization and tracking system in camera sensor networks. *International Journal of Communication Systems* 2014; **27**(4):606–622.
4. Cheng ZY, Liu Y, Chang CC, Guo C. A fault-tolerant group key agreement protocol exploiting dynamic setting. *International Journal of Communication Systems* 2013; **26**(2):259–275.
5. Shi L, Zhang B, Mouftah HT, Ma J. DDRP: an efficient data-driven routing protocol for wireless sensor networks with mobile sinks. *International Journal of Communication Systems* 2013; **26**(10):1341–1355.
6. Liu CX, Liu Y, Zhang ZJ, Cheng ZY. High energy-efficient and privacy-preserving secure data aggregation for wireless sensor networks. *International Journal of Communication Systems* 2013; **26**(3):380–394.
7. Chen H, Deng D, Cheng R, Chilamkurti N, Yu Q. Recent issues in wireless sensor networks. *International Journal of Communication Systems* 2013; **26**(9):1089–1091.
8. Zhang J, Shen X, Zeng H, Dai G, Bo C, Chen F, Lv C. Energy-efficient and localized lossy data aggregation in asynchronous sensor networks. *International Journal of Communication Systems* 2013; **26**(8):989–1010.
9. Wang Y, Shi P, Li K, Chen Z. An energy efficient medium access control protocol for target tracking based on dynamic convey tree collaboration in wireless sensor networks. *International Journal of Communication Systems* 2012; **25**(9):1139–1159.
10. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a Survey. *Computer Networks* 2002; **38**(4):393–422.
11. Chatterjee S, Das AK, Sing JK. Analysis formal security verification of access control schemes in wireless sensor networks: a critical survey. *Journal of Information Assurance and Security* 2013; **8**(1):33–57.
12. Chatterjee S, Das AK, Sing JK. A survey on user access control in wireless sensor networks with formal security verification. *International Journal of Trust Management in Computing and Communications* 2014. In Press.
13. Das AK. A survey on analytic studies of key distribution mechanisms in wireless sensor networks. *Journal of Information Assurance and Security* 2010; **5**(5):526–553.
14. Perrig A, Stankovic J, Wagner D. Security in wireless sensor networks. *Communications of the ACM* 2004; **47**(6): 53–57.
15. Ameen MA, Liu J, Kwak K. Security and privacy issues in wireless sensor networks for healthcare applications. *Journal of Medical Systems* 2012; **36**(1):93–101.
16. Das AK, Sharma P, Chatterjee S, Sing JK. A dynamic password-based user authentication scheme for hierarchical wireless sensor networks. *Journal of Network and Computer Applications* 2012; **35**(5):1646–1656.
17. Chen TH, Shih WK. A robust mutual authentication protocol for wireless sensor networks. *ETRI Journal* 2010; **32**(5):704–712.
18. Fan R, Ping LD, Fu JQ, Pan XZ. A secure and efficient user authentication protocol for two-tieres wireless sensor networks. *Second Pacific-Asia Conference on Circuits, Communications and System (PACCS 2010)*, Beijing, China, 2010; 425–428.
19. He D, Gao Y, Chan S, Chen C, Bu J. An enhanced two-factor user authentication scheme in wireless sensor networks. *Ad Hoc & Sensor Wireless Networks* 2010; **10**(4):361–371.
20. Khan MK, Alghathbar K. Cryptanalysis and security improvements of two-factor user authentication in wireless sensor networks. *Sensors* 2010; **10**:2450–2459.
21. Lee CC, Li CT, Chen SD. Two attacks on a two-factor user authentication in wireless sensor networks. *Parallel Processing Letters* 2011; **21**(1):21–26.

DOI: 10.1002/dac

22. Vaidya B, Makrakis D, Mouftah HT. Improved two-factor user authentication in wireless sensor networks. *Second International Workshop on Network Assurance and Security Services in Ubiquitous Environments*, Niagara Falls, ON, Canada, 2010; 600–606.
23. Wong K, Zheng Y, Cao J, Wang S. A dynamic user authentication scheme for wireless sensor networks. *Proceedings of IEEE International Conference on Sensor Networks, Ubiquitous, Trustworthy Computing*, Vol. 1, Taichung, Taiwan, 2006; 244–251.
24. Das AK, Chatterjee S, Sing JK. Formal security verification of a dynamic password-based user authentication scheme for hierarchical wireless sensor networks. *International Symposium on Security in Computing and Communications (SSCC 2013), Communications in Computer and Information Science Series (CCIS)*, Vol. 377, Mysore, India, 2013; 243–254.
25. Wang D, Wang P. Understanding security failures of two-factor authentication schemes for real-time applications in hierarchical wireless sensor networks. *Ad Hoc Networks* 2014; **20**:1–15. In Press. Available from: http://dx.doi.org/10.1016/j.adhoc.2014.03.003 [Accessed on April 2014].
26. Yuan J, Jiang C, Jiang Z. A biometric-based user authentication for wireless sensor networks. *Wuhan University Journal of Natural Sciences* 2010; **15**(3):272–276.
27. Das ML. Two-factor user authentication in wireless sensor networks. *IEEE Transactions on Wireless Communications* 2009; **8**(3):1086–1090.
28. Das AK, Chatterjee S, Sing JK. A new biometric-based remote user authentication scheme in hierarchical wireless body area sensor networks. *Ad Hoc & Sensor Wireless Networks* 2014. In Press.
29. Althobaiti O, Al-Rodhaan M, Al-Dhelaan A. An efficient biometric authentication protocol for wireless sensor networks. *International Journal of Distributed Sensor Networks* 2013; **2013**:1–13. Article ID 407971, Available from: http://dx.doi.org/10.1155/2013/407971 [Accessed on April 2013].
30. Das AK. Cryptanalysis of an efficient biometric authentication protocol for wireless sensor networks. In *Security in Computing and Communications, Communications in Computer and Information Science*, Vol. 467, Mauri J, Thampi S, Rawat D, Jin D (eds). Springer: Berlin, Heidelberg, 2014; 1–9. Available from: http://dx.doi.org/10.1007/978-3-662-44966-0_1 [Accessed on September 2014].
31. Das AK, Bruhadeshwar B. A biometric-based user authentication scheme for heterogeneous wireless sensor networks. *27th International Conference on Advanced Information Networking and Applications Workshops, WAINA 2013*, Barcelona, Spain, 2013; 291–296.
32. Das AK. A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks. *Peer-to-Peer Networking and Applications* 2014:1–22. Available from: http://dx.doi.org/10.1007/s12083-014-0324-9 [Accessed on August 2013].
33. He D, Kumar N, Chilamkurti N. A secure temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. *International Symposium on Wireless and Pervasive Computing (ISWPC 2013)*, Taipei, Taiwan, 2013; 1–6.
34. Das AK, Goswami A. A secure and efficient uniqueness-and-anonymity-preserving remote user authentication scheme for connected health care. *Journal of Medical Systems* 2013; **37**(3):1–16.
35. Sarkar P. A simple and generic construction of authenticated encryption with associated data. *ACM Transactions on Information and System Security* 2010; **13**(4):33.
36. Stinson DR. Some observations on the theory of cryptographic hash functions. *Designs, Codes and Cryptography* 2006; **38**(2):259–277.
37. Odelu V, Das AK, Goswami A. A secure effective key management scheme for dynamic access control in a large leaf class hierarchy. *Information Sciences* 2014; **269**(C):270–285.
38. Wu S, Chen K. An efficient key-management scheme for hierarchical access control in e-medicine system. *Journal of Medical Systems* 2012; **36**(4):2325–2337.
39. Burnett A, Byrne F, Dowling T, Duffy A. A biometric identity based signature scheme. *International Journal of Network Security* 2007; **5**(3):317–326.
40. Dodis Y, Reyzin L, Smith A. Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. *Proceedings of the Advances in Cryptology (Eurocrypt'04), LNCS*, Vol. 3027, Interlaken, Switzerland, 2004; 523–540.
41. He D, Kumar N, Lee JH, Sherratt RS. Enhanced three-factor security protocol for consumer USB mass storage devices. *IEEE Transactions on Consumer Electronics* 2014; **60**(1):30–37.
42. Standard SH. FIPS PUB 180-1, National Institute of Standards and Technology (NIST), U.S. Department of Commerce, 1995.
43. Kocher P, Jaffe J, Jun B. Differential power analysis. *Proceedings of Advances in Cryptology - CRYPTO'99, LNCS*, Vol. 1666, Santa Barbara, California, USA, 1999; 388–397.
44. Messerges TS, Dabbish EA, Sloan RH. Examining smart-card security under the threat of power analysis attacks. *IEEE Transactions on Computers* 2002; **51**(5):541–552.
45. Dolev D, Yao A. On the security of public key protocols. *IEEE Transactions on Information Theory* 1983; **29**(2):198–208.
46. Chuang YH, Tseng YM. An efficient dynamic group key agreement protocol for imbalanced wireless networks. *International Journal of Network Management* 2010; **20**(4):167–180.
47. Armando A, Basin D, Boichut Y, Chevalier Y, Compagna L, Cuellar J, Drielsma PH, Heám PC, Kouchnarenko O, Mantovani J, Mödersheim S, von Oheimb D, Rusinowitch M, Santiago J, Turuani M, Viganò L, Vigneron L. The avispa tool for the automated validation of internet security protocols and applications. In *17th International*

DOI: 10.1002/dac

Conference on Computer Aided Verification (CAV'05), vol. 3576, Lecture Notes in Computer Science. Springer-Verlag: Edinburgh, Scotland, UK, 2005; 281–285.
48. Automated validation of internet security protocols and applications. Available from: http://www.avispa-project.org/ [Accessed on January 2014].
49. von Oheimb D. The high-level protocol specification language HLPSL developed in the EU project avispa. *Proceedings of APPSEM 2005 Workshop*, Tallinn, 2005; 1–17.
50. Basin D, Modersheim S, Vigano L. OFMC: a symbolic model checker for security protocols. *International Journal of Information Security* 2005; **4**(3):181–208.
51. AVISPA Web Tool. Available from: http://www.avispa-project.org/web-interface/expert.php/ [Accessed on July 2014].
52. Yoo SG, Park KY, Kim J. A security-performance-balanced user authentication scheme for wireless sensor networks. *International Journal of Distributed Sensor Networks* 2012; **2012**:11. Article ID 382810, DOI: 10.1155/2012/382810.
53. Sun DZ, Li JX, Feng ZY, Cao ZF, Xu GQ. On the security and improvement of a two-factor user authentication scheme in wireless sensor networks. *Personal and Ubiquitous Computing* 2013; **17**(5):895–905.
54. Xue K, Ma C, Hong P, Ding R. A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks. *Journal of Network and Computer Applications* 2013; **36**(1):316–323.
55. Jiang Q, Ma J, Lu X, Tian Y. An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks. *Peer-to-Peer Networking and Applications* 2014:1–12. DOI: 10.1007/s12083-014-0285-z.

DOI: 10.1002/dac