




An Efficient, Robust, and Lightweight Subtree-Based Three-Factor Authentication Procedure for Large-Scale DWSN in Random Oracle

Chandrashekhar Meshram , Mohammad S. Obaidat , *Fellow, IEEE*, Cheng-Chi Lee ,
and Sarita Gajbhiye Meshram 

Abstract—Wireless sensor networks (WSNs) are the backbones of numerous real-time monitoring systems that are applied to serve many different parts of our everyday lives including traffic management, telecare, pollution control, military application, among others. In most cases, WSN systems involve exchanges of sensitive/private data between the sensor nodes and the outside world. In order to preserve data privacy, illegal data access must be denied, and so the remote client has to be properly authorized by both the base station and the sensor node in order to ensure data access legitimacy. Many authentication procedures have been projected by researchers based on various frameworks of parameters such as (two-factor authentication (2-FA) = Smart card + Password) and (three-factor authentication (3-FA) = Biometric + Smart card + Password) in the literature. Das *et al.* (2015) projected a three-factor technique for resource-constrained distributed WSN to address the existing pitfalls. In this article, we present an analysis of Das *et al.*'s technique and point out some inconsistencies in the technique; demonstrating that the system is vulnerable against a known session-specific particular information attack, which thus prompts leakage of the client identity. We offer a robust subtree-based 3-FA procedure to fix the problem. In addition, we show the security strengths of our devised approach which have been established both informally and formally using the random oracle model and AVISPA tool.

Manuscript received April 25, 2020; revised August 20, 2020, November 29, 2020, and December 28, 2020; accepted December 29, 2020. The work of Mohammad S. Obaidat was supported by PR of China Ministry of Education Distinguished Possessor under Grant MS2017BJKJ003. (Corresponding author: Chandrashekhar Meshram; Cheng-Chi Lee.)

Chandrashekhar Meshram is with the Department of Postgraduate Studies and Research in Mathematics, Jaywanti Haksar Government Postgraduation College, College of Chhindwara University, Betul 480001, India (e-mail: csmeshram84pdf@gmail.com).

Mohammad S. Obaidat is with the Founding Dean and Professor of College of Computing and Informatics, University of Sharjah, Sharjah 27272, UAE with the King Abdullah II School of Information Technology Department, The University of Jordan, Amman 11942, Jordan, and also with the University of Science and Technology Beijing, Beijing 100083, China (e-mail: msobaidat@gmail.com; s.obaidat@ieee.org).

Cheng-Chi Lee is with the Department of Library and Information Science, Research and Development Center for Physical Education, Health, and Information Technology, Fu Jen Catholic University, New Taipei 24205, R.O.C., and also with the Department of Photonics and Communication Engineering Asia University, Wufeng Shiang, Taiwan 413, R.O.C. (e-mail: clee@mail.fju.edu.tw).

Sarita Gajbhiye Meshram is with the Department for Management of Science and Technology Development, Ton Duc Thang University, Ho Chi Minh City 758307, Vietnam, and also with the Faculty of Environment and Labour Safety, Ton Duc Thang University, Ho Chi Minh City 758307, Vietnam (e-mail: saritagmeshram@tdtu.edu.vn).

Digital Object Identifier 10.1109/JSYST.2021.3049163

Index Terms—Authentication, biometrics, fuzzy extractor, hash function, random oracle, security attacks, smart cards, subtree, wireless sensor networks.

I. INTRODUCTION

THE distributed wireless sensor network (DWSN) is a dynamic infrastructure that consists of lightweight, resource constrained, battery-backup sensor nodes, or motes where communication is performed wirelessly in a smaller scope [1]. In DWSN, sensor nodes are normally randomly located everywhere throughout the objective field to form a multihop wireless communication environment among clients, the base station (BS)/sink node, and sensors. In such a network, as shown in Fig. 1, the BS has unlimited storage capacity and computational resources, and it communicates with the external world over a wireless ad hoc network. The BS monitors and controls the whole network and therefore has the authority to read data from sensors. It is assumed to be trustworthy and not subject to compromise by an attacker.

For the BS to do the job, the most convenient design would be one where the BS serves as a gateway to the WSN with all the client queries routed through it, so that it would be the easiest for the BS to monitor and control the whole system. However, when an emergency case occurs in a system for healthcare monitoring such as forest fire detection or natural disaster prevention, the clients would typically want to have direct access to local sensors rather than getting routed through the BS. To offer a design where direct communication is allowed between clients/users and sensor nodes, there are security issues to take care of regarding the authorization of clients before they connect to the network so as to maintain data privacy, especially in a wireless environment where security threats and possible attackers can be anywhere. To us, the greatest challenge is to design a DWSN system that offers optimum security protection with the least overhead.

In the literature, many remote client authentication procedures have been proposed that use various factors such as two factor authentication (2-FA) [2]–[4], elliptic curve cryptography (ECC) [5], [6], and bilinear pairing [7], [8]. However, some latest research proved that for WSN, biometric-based client authentication is more dependable and secure than conventional password-based client authentication procedures [9]–[14]. Inherent advantages of biometric-based methods include the following statements.

1) Biometric values or keys cannot be forgotten or lost; 2) Biometric values or keys are very difficult to share or copy;

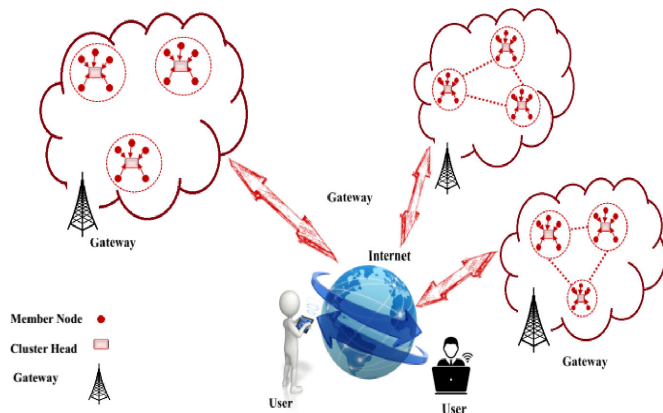


Fig. 1. Distributed wireless sensor network.

3) Biometric values or keys are extremely hard to distribute or forge; 4) Biometric values or keys cannot be easily guessed.

Therefore, in recent years, biometric-based remote client authentication has received much attention with many procedures developed and proposed.

In this article, we show that Das's [15] procedure is vulnerable to several cryptographic attacks and has some internal inconsistencies. To solve the problems pointed out, we have designed an improved, robust three-factor client authentication procedure for DWSN, which is an enhanced version of Das's [15] procedure combined with some other similar procedures.

In 2004, Watro *et al.* [16] offered an authentication procedure for WSN using Diffie–Hellman and RSA-algorithms [17]. As discussed in [18], Watro *et al.*'s [16] procedure has a security problem in that an attacker can have the session key compromised using a client's public key. Then, in 2006, Wong *et al.* [19] presented a password-based authentication procedure, which was advanced shown to be vulnerable against stolen verifier attacks and would also have a problem when multiple clients sign in with the identical login ID.

In 2009, Das [18] projected an improved version of Wong *et al.*'s [19] procedure, claiming that their enhanced procedure was capable of providing secure authentication and session key establishment with high efficiency. However, Vaidya *et al.* [20] proved that Das's [18] procedure was also vulnerable against a stolen smart card attack. As a remedy, Vaidya *et al.* [20] provided an enhanced two-factor client authentication design for WSN. In 2010, He *et al.* [21] projected an enhanced version of Das's [18] procedure. On the other hand, in 2010, Chen and Shih [22] proved that Das's [18] procedure fails to support mutual authentication. At the same time, Chen and Shih [22] projected a robust mutual authentication procedure for WSN. In the same year, Fan *et al.* presented a straightforward and efficient client authentication procedure for two-tiered WSN that can resist denial-of-service (DoS) attacks. Furthermore, in 2010, Yuan *et al.* [23] proposed a procedure that uses biometrics verification alongside password verification for the authentication of the client. Later in 2015, Das [15] proved that Yuan *et al.*'s [23] procedure was weak against DoS attacks and node compromise attacks.

In 2011, Khan *et al.* [24] projected a remote client authentication procedure to be applied on mobile devices, utilizing lightweight hash functions and clients' biometric data (fingerprints). Chen *et al.* [25] demonstrated that Khan *et al.*'s [24]

procedure is vulnerable against fake login attacks through loss of mobile device in 2012. Chen *et al.* [25] then projected an enhanced procedure to fix the problem of Khan *et al.*'s [24] procedure. However, in 2012, Truong *et al.* [26] proved that Chen *et al.*'s [25] procedure is weak against the replay attack and fails to support client anonymity. On demonstrating the vulnerabilities in Chen *et al.*'s [25] procedure, Truong *et al.* [26] presented an enhanced procedure. Later in 2014, Khan *et al.* [27] indicated that both Chen *et al.*'s [25] procedure and Truong *et al.*'s [26] procedure have their security flaws, and so they offered an enhanced procedure to fix the problems.

In 2016, Amin and Biswas [28] projected a secure authentication procedure for multigateway wireless sensor networks. Recently, Sharif *et al.*'s [29] introduced a secure and lightweight authentication and key agreement protocol for Internet of things-based WSNs that is free from the security challenges of previous procedures. Mood and Nikooghadam [30] introduced an authentication and key agreement scheme based on elliptic curve cryptography that are secure against the well-known attacks, and also provide the perfect forward secrecy and introduced its new variant for smart grid communications [31]. Mood and Nikooghadam [32] introduced an anonymous password-authenticated key exchange procedure using extended Chebyshev chaotic maps that provides the user anonymity.

A. Motivation

While some researchers have proposed security mechanisms, they are not lightweight enough to meet the WSN system's needs. In this article, we have proposed an efficient secure and lightweight procedure which offers mutual authentication and key agreement for large-scale DWSN in random oracle.

Recently, Das [15] projected a novel three-factor client authentication procedure for DWSN, where the clients can well refresh their passwords and biometric data without contacting the BS. Das also showed that the [15] procedure was protected against various known attacks. In this article, we shall point out that Das's [15] procedure still has security flaws in consistency as well as synchronization and is weak against the replay attack as well as known session particular temporary information attack. Besides, Das's [15] procedure fails to preserve client anonymity, and it has some feasibility issues. To solve all these issues, we demonstrate a new procedure that is more robust and more efficient. In addition, this article introduces a new lightweight 3-FA and key agreement procedure for the large-scale DWSN to enhance the security of the previous procedures. The protection of the proposed procedure is tested using the simulator tool automated validation of internet security protocols and applications (AVISPA) and fully addressed in an informal manner.

B. Contributions

The contributions of this article are summarized as follows.

1) The proposed procedure offers the best cost of storage relative to the relevant procedures; 2) The key innovation of our article is that while retaining its high efficiency, the proposed procedure offers the perfect forward secrecy; 3) We devise a robust and efficient subtree-based lightweight client authentication procedure best suited for DWSN instead of using cryptographic operations like encryption and decryption that are far more computationally demanding, which is the case with Das's [15] procedure; 4) The presented procedure is user-friendly in that it

provides the client with the power to directly modify/update his password and personal biometric key without having to contact the BS; 5) Thorough security examinations have demonstrated that our new procedure is protected against various recognized attacks.

The strengths of the procedure are the following:

1) the presented procedure offers the perfect forward secrecy; 2) the proposed procedure offers the best cost of energy storage relative to the relevant procedures; 3) the presented procedure is user-friendly in that it provides the client with the power to directly modify/update his password and personal biometric key without having to contact the BS.

The weakness is

1) because of using smartcard, our method also needs a card reader to verify a login user in login stage.

In Section II, we provide the basic concepts and notations we use in our new procedure. In Section III, we briefly introduce Das's [15] procedure; then, in Section IV, we present our evaluation of Das's [15] procedure to show where the vulnerabilities are. After that, in Section V, we present the details of our new procedure. In Section VI, we discuss the security examination of improved procedure. In Section VII, we show the results of a formal security evaluation of our new procedure that we have conducted using the random oracle model. Then, in Section VIII, we present a couple of tables and show how our new procedure compares with some associated procedures in terms of some security assets as well as computation cost. Lastly, Section IX concludes the article.

II. BACKGROUND MATERIALS

In this section, we shall first define some notations we use in our new procedure that are about fuzzy-entity data sharing, and then we will specify the concept of fuzzy extractor. Table I shows the symbols utilized in the suggested procedure.

A. Symbolizations

A robust lightweight 3-FA procedure for large-scale DWSN is a novel cryptographic primitive for fuzzy-entity data sharing. Let us see how some notations are defined, as they will later be used when we get to the details of our new procedure. We utilize $[a, b]$ for the shorthand of $\{a, a+1, \dots, b\}$, and $[a]$ for $[1, a]$ if no uncertainty is presented. For every $id = (id_1, id_2, \dots, id_d)$, where id is an identity vector, let $S_{id} = \{id_1, \dots, id_d\}$ be a set identities performing in the form of id . We define $I_{id} = \{i : id_i \in S_{id}\}$ as the position records of id in the tree structure of the framework. The projected [33] receivers in our new procedure are portrayed as a subtree \mathbb{T} . The identity vectors and their receivers' positions in the tree structure are incorporated into \mathbb{T} . We require that any legitimate \mathbb{T} must contain the root node. This reflects the fact that the framework is managed by the private key generator (PKG).

Similarly, the identity set of \mathbb{T} and the position indices of \mathbb{T} are represented by $S_{\mathbb{T}} = \cup_{id \in \mathbb{T}} S_{id}$ and $I_{id} = \{i : id_i \in S_{\mathbb{T}}\}$, respectively. The notations here can be expressed as $\text{Sup}(id) = \{(id_1, id_2, \dots, id_{k'}) : k' \leq k\}$ to illustrate the superiors of $id = (id_1, id_2, \dots, id_k)$. The intended receivers represented in the subtree \mathbb{T} are characterized as $\text{Sup}(\mathbb{T}) = \cup_{id \in \mathbb{T}} \text{Sup}(id)$, respectively.

TABLE I
LIST OF NOTATIONS USED IN THE PROPOSED PROCEDURE

Symbols	Descriptions
C_i	The i^{th} client
ID_i	Unique identity of C_i .
id_i	Super identity of C_i ; $id \in \text{Sup}(\mathbb{T})$.
PW_i	Unique password of C_i .
B_i	Biometric information of id_i .
T_i	Current system time stamp.
ΔT	Maximum transmission delay.
SC	Smart card.
BS	Base station
SN_j	Sensor node ' j '.
SID_j	Identity of sensor ' j '. (we use the term SID_j instead of $IDSN_j$)
Sid_j	Super identity of sensor ' j '. (we use the term Sid_j instead of $idSN_j$)
MK_{SN_j}	Master key shared between BS and SN_j .
$PBIO_i$	Personal biometric data of client C_i .
X_s	1024-bit secret master key of BS .
RN_x	A arbitrary number created by client ' x '.
$E_k(M)$	Symmetric encryption process of M (message) using key ' k '.
$D_k(M)$	Symmetric decryption process of M (message) using key ' k '.
$h(\cdot)$	One way hash function.
$H(\cdot)$	Bio-hash function.
\oplus	Ex-OR operation.
$Gen(B_i)$	Fuzzy Extractor function.
σ_i	Biometric data key
NL_{SN_j}	List of neighboring nodes
$Gen(\cdot)$	Fuzzy extractor function

B. Fuzzy Extractor

The biometric info is susceptible to noise contamination amid data attainment, and therefore the reproduction of the original biometric data is extremely difficult. To escape this issue, a fuzzy extractor [4], [5] has been preferable.

Definition: The fuzzy extractor is a tuple (M, l, t) characterized by two accompanying algorithms known as Gen and Rep: i) $\text{Gen}(B_i) = \{\sigma_i, \tau_i\}$: It is a probabilistic procedure that takes some biometric data $B_i \in M$ as info and yields a secret data key $\sigma_i \in \{0, 1\}^l$ and a public propagation parameter τ_i . ii) $\text{Rep}(B_i^*, \tau_i) = \sigma_i$, such that $d(B_i, B_i^*) \leq t$: It takes in some loud biometric data $B_i^* \in M$ and a public parameter τ_i identified with B_i , and after that it repeats the biometric secret data key σ_i .

III. ASSESSMENT OF DAS'S PROCEDURE

In this section, we examine Das's [15] 3-FA procedure for large scale DWSNs. Das's [15] procedure includes seven stages, which are the predeployment stage, the postdeployment stage, the registration stage, the login phase, the authentication and key agreement stage, the password and biometric update stage, and finally the dynamic node addition stage. The symbols utilized in Das's [15] and Table I procedure are listed below.

A. Predeployment Stage

The motivation behind this stage is for the *BS* to be able to assign identities and keys to all sensors offline.

- (A1) *BS* allots (SID_j) a novel identity to every conveyed sensor in the DWSN.
- (A2) For every sensor, *BS* arbitrarily creates an exclusive SMK_j master key.
- (A3) For every conveyed sensor, *BS* loads its (SID_j) identity and SMK_j master key.
- (A4) *BS* also stocks the identity and master key for each deployed sensor.

B. Postdeployment Stage

This stage is executed after all sensors are laid in place.

- (B1) Every sensor node keeps a list of neighboring nodes $NL_{SN_j} = \{SN_{j1}, SN_{j2}, \dots, SN_{jd}\}$.
- (B2) Each sensor (SN_j) can use its own identity (SID_j) to broadcast a sample message saying “OK” to entirely the neighboring nodes in its range.
- (B3) Each sensor node obtains a trial message saying “OK” from each of its neighbors.

C. Registration Stage

This stage is performed between a client and the *BS*, and only a registered client can later login and access a sensor node to obtain local data within DWSN.

(C1) Client picks an ID_i , a password PW_i , and a 1024-bit random number K .

(C2) Client inputs biometric information B_i to $Gen(\cdot)$ fuzzy extractor function. This will yield $Gen(B_i) = (\sigma_i, \tau_i)$, where σ_i is the biometric data key and kept it secret to client, and τ_i is the public parameter.

(C3) Client calculates $RPW_i = h(ID_i || K || PW_i)$ and sends a registration appeal to *BS* through a protected channel.

(C4) After getting the registration appeal, *BS* produces its secret key X_S and processes $r_i = h(ID_i || X_S)$.

(C5) *BS* $\rightarrow U_i$ [Smartcard $SC = \{r_i, h(\cdot)\}$] over a secure channel.

(C6) Upon receiving SC , C_i computes:

$$e_i = h((ID_i || \sigma_i \oplus K), f_i = h(ID_i || RPW || \sigma_i)$$

$$r_i^* = r_i \oplus h(ID_i || K) = h(ID_i || X_S) \oplus h(ID_i || K).$$

(C7) C_i replaces r_i with r_i^* in SC . Now SC contains $\{e_i, f_i, r_i, r_i^*, Gen(\cdot), h(\cdot), Rep(\cdot)\}$.

D. Login Stage

This stage is performed when a registered client wants to access a sensor node and obtain local data from DWSN.

(D1) Client inserts her/his SC into the card reader, inputs her/his pair of ID_i and PW_i , and also provides his/her B_i^* by passing it through the sensor.

(D2) Now SC computes: $\sigma_i^* = Rep(B_i^*, \tau_i)$; $K^* = h(ID_i || \sigma_i^*) \oplus e_i$; $RPW_i^* = h(ID_i || K^* || PW_i)$; $f_i^* = h(ID_i || RPW_i^* || \sigma_i)$.

(D3) Then SC compares $f_i^* = ? f_i$. If yes, then the client authentication is successful, and this stage is finished.

(D4) After client authentication, C_i chooses a sensor node SN_j from which she/he needs to obtain data, and also C_i chooses an arbitrary number RNC_i .

(D5) SC calculates $M_1 = r_i^* \oplus h(ID_i || K^*) = h(ID_i || X_S)$; $M_2 = M_1 \oplus RNC_i = h(ID_i || X_S) \oplus RNC_i$; $M_3 = h(ID_i || SID_j || M_1 || RNC_i || T_1)$, and drives $\{SID_j, M_3, M_2, T_1\}$ as the login request message to *BS* through a public network, where T_1 is the time of sending the login request.

E. Authentication and Key Agreement Stage

In this stage, the *BS* authenticates a client, and the sensor node authenticates the *BS*. These are done so that a common session key can be established between the client and the sensor, allowing the client to access data from a sensor node.

(E1) After getting the login appeal from C_i at time T_2 , *BS* first confirms the time legitimacy by checking whether $T_2 - T_1 < = \Delta T$. If the time is not legal, then the login appeal is rejected immediately.

(E2) If the time is legitimate, *BS* calculates $h(ID_i || X_S)$; $M_5 = M_2 \oplus M_4 = RNC_i$; $M_6 = h(ID_i || SID_j || M_4 || M_5 || T_1)$; and verifies ($M_6 = M_3$) for client authentication. If the verification is not successful, then this stage is finished.

(E3) If M_6 checks out, *BS* computes an encrypted message $M_7 = E_{SMK_j} [ID_i, SID_j, h(M_4), M_5, T_3, T_1]$ using the master key of SN_j , where T_3 is the current time stamp of *BS*. Lastly, *BS* sends an authentication appeal $\{M_7, SID_j\}$ to the sensor node through a public network.

(E4) Upon getting the authentication message at time T_4 , sensor decrypts this message using SMK_j master key and confirms time legitimacy as $T_4 - T_3 < = \Delta T$. If the time is valid, then the message is legitimate.

(E5) Now sensor chooses an arbitrary number RNS_j and generates a session key SK_{ij} as $SK_{ij} = h(ID_i || SID_j || h(M_4) || M_5 || RNS_j || T_1 || T_5)$ that will be shared only between client and sensor, where T_5 is the sensor's present time stamp.

(E6) Sensor node SN_j also calculates $M_8 = h(SK_{ij})$; $M_9 = M_5 \oplus RNS_j \oplus ID_i = RNC_j \oplus RNS_j \oplus ID_i$ and then drives a login response message $\{M_9, M_8, T_5\}$ to the client U_i through a public network.

(E7) Upon getting the login response message at time T_6 , SC of C_i first confirms time legitimacy by checking whether $T_6 - T_5 < = \Delta T$. If the time is not valid, then the login request is rejected immediately.

(E8) If T_6 checks out, SC_i then computes $M_{10} = M_9 \oplus RNC_j \oplus ID_i = RNS_j$; $SK_{ij}^* = h(ID_i || SID_j || h(M_1) || RNC_j || M_{10} || T_1 || T_5)$; $M_{11} = h(SK_{ij}^*)$ and verifies ($M_{11} = M_8$) for sensor authentication. If the verification is successful, this SK_{ij}^* will be used between client and sensor for further secure communication, and this concludes the authentication and key agreement stage.

F. Password and Biometric Update Stage

In this stage, a legal client can amend her/his password and biometric info without the involvement of *BS* as follows.

(F1) The client inserts her/his SC into the card reader and inputs her/his ID_i , old PW_i , and then the client inputs his/her old B_i^* by passing it through the sensor.

(F2) Now SC computes $\sigma_i^{old} = Rep(B_i^{old}, \tau_i)$; $K^* = h(ID_i || \sigma_i^{old}) \oplus e_i$; $RPW_i^{old} = h(ID_i || K^* || PW_i^{old})$; $f_i^{old} = h(ID_i || RPW_i^{old} || \sigma_i^{old})$.

(F3) Then SC compares ($f_i^{\text{old}} = f_i$). If the verification is successful, then SC prompts client to enter his/her new PW_i and new B_i .

(F4) Then SC computes: $(\sigma_i^{\text{new}}, \tau_i^{\text{new}}) = \text{Gen}(B_i^{\text{new}})$; $e_i^{\text{new}} = h(ID_i || \sigma_i^{\text{new}}) \oplus K$; $RPW_i^{\text{new}} = h(ID_i || K^* || PW_i^{\text{new}})$; $f_i^{\text{new}} = h(ID_i || RPW_i^{\text{new}} || \sigma_i^{\text{new}})$.

(F5) Finally SC updates e_i with e_i^{new} , f_i with f_i^{new} and τ_i with τ_i^{new} in its memory.

G. Dynamic Node Addition Stage

This stage allows the adding of new sensor nodes and also the replacement of old/defective/compromised sensors with new ones in DWSN.

(G1) The procedure in this stage is performed by BS offline.

(G2) For the addition of a new sensor, BS assigns SID_j^{new} and chooses a unique master key SMK_j^{new} .

(G3) BS preloads SID_j^{new} and SMK_j^{new} into the memory of SN_j^{new} before laying it in place.

(G4) BS informs all clients about the newly added sensor (SN_j^{new}) so they can utilize its services.

IV. SECURITY EXAMINATION OF DAS'S PROCEDURE

In this section, we will show that Das's [15] procedure has some security flaws including a consistency problem, failure to support client anonymity, synchronization trouble, risks of random number leakage, a feasibility problem, and vulnerability to the replay attack. These will be discussed below.

A. Threat Model

In this article, we will adopt the comparative threat model used by Das [15] and other researchers [2]–[8], [12]. 1) A foe or legal client can remove the data stored in the smart card one way or another, for example, using power consumption or leakage of information means. 2) A foe is presumably able to screen or eavesdrop on all the login requests as well as login response messages exchanged between client and BS over the public channel, namely, the Internet. 3) A foe can change, remove, replay/resend, and/or forward the eavesdropped messages. A foe can be a genuine client or an outsider in any framework.

B. Inconsistency Problem

In the login stage of the Das [15] procedure, the client U_i/SC_i sends in a login appeal $\{SID_j, M_3, M_2, T_1\}$ to the BS without presenting the client identity ID_i . The BS cannot tell which client sent in this login request because there is no way to compute ID_i by using any value obtained from the received login request. Normally there will be times when many registered clients send in their login requests at the same time, which will cause an inconsistency problem.

C. Weakness Against the Known Session-Specific Temporary Information Attack

Proper protection against the attack should guarantee that all the session keys be protected even if the session specific random numbers are known to a foe E . According to [34]–[36], protection against this attack is important, and this type of

attack is likely to happen and cause damage under the following circumstances [37], [38].

1) If the sensor and client trust in some external or internal random number generator that may be compromised by foe E ;
2) During each session the random numbers are stored in the device. If the numbers remain unerased, then foe E might access them by taking control of the client's device or the sensor. As explained above, a foe who can compromise session-specific random numbers (RNC_i, RNS_j) can frame the session key $\{SK_{ij} = h(ID_i || SID_j || h(M_4) || M_5 || RNS_j || T_1 || T_5)\}$ as follows.

a) Suppose M_5 in the session key, i.e., the client selected random number RNC_i , is compromised by the attacker; b) The foe is now able to obtain the client's identity (ID_i), sensor's identity (SID_j), and the time stamps (T_1 and T_5) directly by observing the messages exchanged between the client, the BS, and the sensor via public channels; c) The foe can calculate M_4 from $M_2 = \{M_1 \oplus RNC_i\}$, which is available by observing the login request message. With RNC_i compromised, the attacker can compute $M_1 = M_2 \oplus RNC_i$; $M_1 = M_4 = h(ID_i || X_S)$; d) Now the foe can easily structure the session key because all the required values are ready. This means Das's [15] procedure is weak against the known session-specific random number attack.

D. Vulnerability to the Replay Attack

A replay attack happens when the attacker captures some message that is being transmitted among the client, the BS, and the sensor, and after that tries to impersonate a genuine client by retransmitting this captured message. These types of attack can definitely take effect on Das's [15] procedure when the attacker captures and resends a login appeal message to BS inside the time limit ΔT . BS would not be able to identify whether it is the original message or a retransmitted message by a foe. That implies BS will receive two back to back login appeal messages inside the time limit ΔT : One from the client and the other from the attacker. The original message sent by the client will be processed successfully. On the other hand, to respond to the replayed message sent by the foe, BS will send an encrypted message $\{SID_j, M_7\}$ to the sensor, and the sensor will decrypt this message, do some corresponding computation, and send a login replay message $\{M_9, M_8, T_5\}$ to the attacker. Although the foe cannot use this message to calculate the session key, but this extra work will result in additional overhead for both BS and the sensor, which will then affect the overall performance of the system.

V. PROPOSED ENHANCED PROCEDURE

A. Predeployment Stage

The motivation behind this stage is for the BS to preassign the keys to all sensors offline before the sensors are laid in place.

(PR1) BS allots (Sid_j) an original identity (Sid_j) for every conveyed sensor in the DWSN.

(PR 2) For every sensor, BS arbitrarily creates an exclusive SMK_j master key.

(PR 3) For each conveyed sensor, BS loads its (Sid_j) identity and (SMK_j) master key.

(PR 4) BS also stores the identity and master key for each deployed sensor.

B. Postdeployment Stage

This stage is executed when all sensors have been located into their designated locations.

(PO1) Each sensor node keeps a list of neighboring nodes $NL_{SN_j} = \{SN_{j1}, SN_{j2}, \dots, SN_{jd}\}$.

(PO2) Each sensor (SN_j) broadcasts a sample message saying “OK” with its own (Sid_j) identity to every neighboring nodes in its range.

(PO3) Each sensor node gets a sample message saying “OK” from each of its neighbors.

C. Registration Stage

This stage is summoned at whatever point of time when a client C_i wants to register with the BS. The procedure includes the following steps.

(R1) Client first picks his/her id_i , password PW_i , and a 1024-bit random number “ $n = q.p$,” where q and p are huge prime numbers.

(R2) Client also provides his/her biometric info B_i and inputs it to the Gen(.) fuzzy extractor function, producing $Gen(B_i) = (\sigma_i, \tau_i)$, where σ_i and τ_i are the biometric data keys which are kept secret to the client and the public parameter, respectively.

(R3) Client computes $Rid_i = h(id_i||n)$; $RPW_i = h(id_i||\sigma_i||PW_i||n)$ and sends a registration request $\{Rid_i, RPW_i, n\}$ to BS over a secure channel.

(R4) Upon getting the registration appeal, BS produces its “ X_s ” secret key, picks “ e_i ,” which is identical to every client, and calculates $Z_i = h(Rid_i||X_s||n)$; $c_i = Z_i \oplus RPW_i$; $f_i = h(Rid_i||n||RPW_i)$.

(R5) For each client, BS stores $h(e_i||X_s)$; $G_i = n \oplus h(e_i||X_s)$ in its (database) DB and keeps r_1 (random number), which is sent by the client in every login appeal in a list.

(R6) $BS \rightarrow C_i$ [Smartcard $SC = \{f_i, c_i, e_i\}$] through a secure network.

(R7) Upon getting SC, C_i calculates E_i and N as follows: $E_i = e_i \oplus h(id_i||\sigma_i||PW_i)$; $N_i = n \oplus h(id_i||\sigma_i||PW_i)$

(R8) Now C_i replaces “ e_i ” with E_i and adds N_i, τ_i to SC. Lastly, SC covers $\{f_i, c_i, E_i, N_i, \tau_i, h(\cdot)\}$

D. Login Stage

This stage is to be performed at whatever time a client wants to access info from any sensor in DWSN.

(L1). The client inserts her/his SC into the card reader, inputs her/his id_i, PW_i , and also passes his/her B_i^* through the sensor.

(L2) Now SC computes: $\sigma_i^* = Rep(B_i^*, \tau_i)$, $e_i = E_i \oplus h(id_i||\sigma_i^*||PW_i)$, $n = N_i \oplus h(id_i||\sigma_i^*||PW_i)$, $Rid_i = h(id_i||n)$, $RPW_i = h(id_i||\sigma_i^*||PW_i||n)$, and $f_i^* = h(Rid_i||n||NPW_i)$.

(L3) Then SC compares f_i^* with the present f_i in it. If both are identical, then client authentication is effective, and generally this stage is ended promptly.

(L4) Next is client authentication. The client picks a sensor node SN_j from which she/he needs to access information and furthermore chooses two arbitrary numbers r_1 and r_2 .

(L5) SC computes $Did_i = id_i \oplus h(e_i \oplus n) \oplus r_1$; $Z_i = c_i \oplus RPW_i$, $M_1 = Z_i^* \oplus r_2$; $M_2 = H(id_i||Sid_j||r_1||r_2||e_i||Z_i)$ and finally sends the login appeal message $\{Sid_j, Did_i, M_1, M_2, r_1, e_i\}$ to the BS through a public network.

E Authentication and Key Agreement Stage

In this stage, the BS authenticates the client, and the sensor node authenticates the BS before the client is allowed to access data from the sensor node and to establish a common session key shared between the client and the sensor. (A1) Upon getting the login appeal from C_i , BS first calculates $h(e_i||X_s)$ to find C_i 's info in its DB and then computes $h(e_i||X_s)$ and extracts the value “ n ” by calculating $n = G_i \oplus h(e_i \oplus X_s)$. Then BS stores the usual r_1 random number in its DB.

(A2) BS computes $id_i = Did_i \oplus h(e_i \oplus n) \oplus r_1$; $Rid_i = h(id_i||n)$, $Z_i^* = h(Rid_i||X_s||n)$, $r_2 = M_1 \oplus Z_i^*$, $M_2 = h(id_i||Sid_j||r_1||r_2||e_i||Z_i^*)$ and compares M_2^* with M_2 in the SC for client authentication. If the verification turns out positive, then this stage will be concluded immediately.

(A3) BS picks an arbitrary number “ r_3 ” and further calculates $SDid_j = (id_i \oplus Sid_j) \oplus h(SMK_j||r_3||n)$; $N_j = n \oplus h(SMK_j||r_3)$, $M_3 = h(id_i||Sid_j||Z_i^*||n||SMK_j||r_3)$; $Z_j = Z_i^* \oplus h(r_3||n||SMK_j||Sid_j||id_i)$; $R_j = r_2 \oplus h(|SMK_j||id_i||r_3)$. Finally, BS sends the authentication appeal $\{SDid_j, M_3, N_j, R_j, Z_j, r_3, h(\cdot)\}$ to the sensor node through a public network.

(A4) Upon getting the authentication message, the sensor first calculates $n = N_j \oplus h(SMK_j||r_3)$; $(id_i \oplus Sid_j) = SDid_j \oplus h(SMK_j||r_3||n)$; $id_i = Sid_j \oplus (id_i \oplus Sid_j)$; $Z_i = Z_j \oplus h(r_3||n||SMK_j||Sid_j||id_i)$; $M_3^* = h(id_i||Sid_j||Z_i^*||n||SMK_j||r_3)$ and then verifies the computed M_3^* by checking if it is same as the received M_3 . In the event that yes, the sensor node authenticates BS effectively; else the session will be finished immediately.

(A5) Now the sensor node picks a random number “ r_4 ” and then calculates $r_2 = R_j \oplus h(SMK_j||id_i||r_3)$ and frames a session key $SK_{ij} = h(id_i||r_2||Sid_j||h(Z_i^*)||r_3||r_4)$.

(A6) The sensor node SN_j also computes $M_4 = h(SK_{ij}||r_4||id_i||Sid_j)$; $S_{j1} = r_4 \oplus h(id_i||r_2||Z_i)$;

$S_{j2} = r_3 \oplus h(id_i||Z_i||r_4)$ and then drives a login response $\{M_4, S_{j1}, S_{j2}\}$ message to the client C_i through a public network.

(A7) Upon getting the login response message, SC of C_i calculates $r_4^* = S_{j1} \oplus h(id_i||r_2||Z_i)$; $r_3^* = S_{j2} \oplus h(id_i||Z_i||r_4)$; $SK_{ij}^* = h(id_i||r_2||Sid_j||h(Z_i)||r_3^*||r_4^*)$; $M_4^* = h(SK_{ij}^*||r_4^*||id_i||Sid_j)$, and verifies if $M_4^* = M_4$ for sensor and BS authentication. If M_4^* checks out, this SK_{ij}^* will be used between client and sensor for advance safe communication, and this concludes the stage.

F. Password and Biometric Update Stage

In this stage, a legal client can modified her/his password and biometric record without the involvement of BS as follows.

(P1) The client inserts her/his SC into the card reader and inputs her/his id_i , old PW_i^* , and also passes her/his old B_i^* through the sensor.

(P2) Now SC computes: $\sigma_i^* = Rep(B_i^*, \tau_i)$; $e_i = E_i \oplus h(id_i||\sigma_i^*||PW_i^*)$; $n = N_i \oplus h(id_i \oplus \sigma_i^* \oplus PW_i^*)$; $Rid_i = h(id_i||n)$; $RPW_i = h(id_i||\sigma_i^*||PW_i^*||n)$; $f_i^* = h(Rid_i||n||RPW_i)$.

(P3) Then SC compares f_i^* with f_i , which is present in the SC. If the two are equal, then SC prompts the client to enter his/her new PW_i and new B_i .

(P4) Then SC computes: $Gen(B_i^{new}) = (\sigma_i^{new}, \tau_i^{new})$; $Z_i = c_i \oplus RPW_i$; $RPW_i^{new} = h(id_i || \sigma_i^{new} || PW_i^{new} || n)$; $c_i^{new} = Z_i \oplus RPW_i^{new}$; $f_i^{new} = h(Rid_i || n || RPW_i^{new})$; $E_i^{new} = e_i \oplus h(id_i || \sigma_i^{new} || PW_i^{new})$; $N_i^{new} = n \oplus h(id_i \oplus \sigma_i^{new} \oplus PW_i^{new})$.

(P5) Finally, SC updates its old values c_i , f_i , E_i , N_i , τ_i with c_i^{new} , f_i^{new} , E_i^{new} , N_i^{new} , τ_i^{new} , respectively, in its memory.

G. Dynamic Node Addition Stage

This stage allows the addition of new sensor nodes and the replacement of old/defective/compromised sensor nodes with new ones in DWSN.

(D1) The steps are performed by *BS* offline.

(D2) For the addition of a new sensor, *BS* assigns Sid_j^{new} and chooses a unique master key SMK_j^{new} .

(D3) Next *BS* preloads Sid_j^{new} and SMK_j^{new} into SN_j^{new} 's memory before laying it in place.

(D4) *BS* informs all clients about the addition of this new sensor (SN_j^{new}) so they can utilize its services.

VI. SECURITY EXAMINATION OF IMPROVED PROCEDURE

A. Resistance Against the Stolen Smart Card attack

Numerous researchers have claimed that the information present in the SC can be acquired by using certain procedures such as power consumption analysis, etc. [2], [11], [12]. Here, let us assume that an attacker E robs C_i of his/her SC and acquires the information $\{c_i, f_i, E_i, N_i, \tau_i, h(\cdot)\}$, where $c_i = Z_i \oplus RPW_i$; $f_i = h(Rid_i || n || RPW_i)$; $E_i = e_i \oplus h(id_i || \sigma_i || PW_i)$; and $N_i = n \oplus h(id_i || \sigma_i || PW_i)$. Attacker E cannot speculate the client's password using the above equations since he/she does not have the client's id_i , PW_i , Z_i , σ_i , and the n values. If the attacker knew all values except PW_i , then he/she might try guessing and might succeed; however, guessing more than one value (i.e., $h(Rid_i || n || RPW_i)$, or $e_i \oplus h(id_i || \sigma_i || PW_i)$, or $n \oplus h(id_i || \sigma_i || PW_i)$) at the same time is not possible. Therefore, we can say that our procedure is protected against the lost/stolen smartcard attack.

B. Resistance Against the Online Dictionary Password Guessing Attack

In an online password guessing attack, the foe attempts to login to the server by guessing some distinctive client id_i , password, and biometric information from a lexicon. However, to break our system, the foe has to speculate more than one value (i.e., $h(id_i || \sigma_i || PW_i)$ or $n \oplus h(id_i || \sigma_i || PW_i)$, or $h(Rid_i || n || RPW_i)$) simultaneously, which is not possible. In fact, even guessing one single value within polynomial time (i.e., Δt) is not possible. Let alone guessing biometric information, which is under the protection of each client's unique b_i value. Moreover, the attacker only has few chances (maximum 3) to guess the client's sup id and password because the SC gets locked up if the number of unsuccessful attempts exceeds the maximum count. With the above protections, our new procedure is safe against the on-line password guessing attack.

C. Resistance Against the Replay attack

In this kind of attack, the foe E , first monitors the exchange of messages among the legitimate client, the stations, and the sensor, and then tries to impersonate the client and login to the BS by resending a captured message. Answering a login message of one session into another session is easily detected with our procedure, since the r_1 value is already present in the legitimate client's login message and is already stored in the BS's database. Because the r_1 value can be found in BS's database, the login message will be determined to be a replayed message, and this session will be terminated. Therefore, we can say that our enhanced procedure is secure against the message replay attack.

D. Preservation of Client Anonymity

Client anonymity is well preserved when the client's id_i maintains secure; otherwise, an attacker would be able to speculate the client's password by following the hint of the client's identity. In our proposed procedure, the client's sup id is never transmitted via any public communication channel, so there is no way an attacker can obtain id_i . Therefore, our new procedure can indeed preserve client anonymity.

E. Resistance Against the Known Session-Specific Temporary Information Attack

It is essential that all the session keys be well protected so as to keep the system secure even if the session specific random numbers are known to a foe E . In our new procedure, the session key is $\{SK_{ij} = h(id_i || r_2 || Sid_j || h(Z_i^*) || r_3 || r_4)\}$, where $Z_i = h(Rid_i || X_s || n)$. Here, even if the foe E knows all the arbitrary numbers (r_4 , r_3 , r_2 , and r_1), he/she still cannot frame the session key because he/she does not have id_i , n , X_s , and there is no way E can retrieve these values. Therefore, our devised procedure is secure against the known session-specific random number attack.

F. Practicality in Environments of Lightweight Devices

In our procedure, we use only low computation operations like Ex-OR, concatenation and hash functions instead of encryption and decryption operations to avoid causing too much burden on the sensors, which are lightweight devices. Therefore, our new procedure is practical and applicable.

G. Resistance Against the DoS Attack

In this type of attack, a foe somehow gets hold of the SC of a valid client and then tries to modify the verification information put away on the SC so that the valid client U_i will be unable to sign into the wireless network in ensuing sessions. In our proposed procedure, the SC always checks the legitimacy of the client before any modification can be done. Without the knowledge of the legitimate client's id_i , PW_i and b_i , a foe has no way to pass the authentication so as to update the existing SC values. Therefore, our new procedure is protected against the DoS attack.

Registration Stage								
C_i (Clint)/ smartcard		BS						
C_i selects $id_i, PW_i, B_i, 1024$ bit random number ' n '. Computes $Gen(B_i) = (\sigma_i, \tau_i)$; $Rid_i = h(id_i n)$; $RPW_i = h(id_i \sigma_i PW_i n)$	$\{RID_i, RPW_i, n\}$ Secure channel $SC = \{c_i, f_i, e_i\}$	Choose ' e_i ' unique to each user Computes $Z_i = h(Rid_i X_s n)$, $c_i = Z_i \oplus RPW_i, f_i = h(Rid_i n RPW_i)$ <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>$h(e_i X_s)$</td> <td>$G_i, n,$</td> <td>List r_1 of values</td> </tr> <tr> <td>.....</td> <td>.....</td> <td>.....</td> </tr> </table>	$h(e_i X_s)$	$G_i, n,$	List r_1 of values
$h(e_i X_s)$	$G_i, n,$	List r_1 of values						
.....						
$E_i = e_i \oplus h(id_i \sigma_i PW_i)$, $N_i = n \oplus h(id_i \sigma_i PW_i)$ Replaces ' e_i ' with E_i and adds N_i, τ_i to the SC. Finally, $SC = \{e_i, f_i, \tau_i, r_1^*, Gen(\cdot), Rep(\cdot), h(\cdot)\}$								
Login Stage								
Enters id_i, PW_i, B_i $\sigma_i^* = Rep(B_i^*, \tau_i)$, $e_i = E_i \oplus h(id_i \sigma_i PW_i)$, $n = N_i \oplus h(id_i \sigma_i PW_i)$, $Rid_i = h(id_i n)$, $RPW_i = h(id_i \sigma_i PW_i n)$, $f_i^* = h(Nid_i n RPW_i)$. Verifies $f_i^* =? f_i$ for user authentication SC generates random numbers r_1 and r_2 Computes: $Did_i = id_i \oplus h(e_i \oplus n) \oplus r_1, Z_i = c_i \oplus RPW_i, M_1 = Z_i^* \oplus r_2,$ $M_2 = H(id_i Sid_j r_1 r_2 e_i Z_i)$	$\{Did_i, Sid_j, M_1, M_2, e_i, r_1\}$ via public channel							
Authentication Stage								
BS	SN (S)	Clint (C_i)						
Compute $H(e_i X_s)$ to indexes the DB and retrieve $G_i = n \oplus h(e_i X_s)$ computes $id_i = Did_i \oplus h(e_i \oplus n) \oplus r_1, Rid_i = h(id_i n)$, $Z_i^* = h(Rid_i X_s n), r_2 = M_1 \oplus Z_i^*$, $M_2^* = h(id_i Sid_j r_1 r_2 e_i Z_i^*)$ Verifies $M_2^* =? M_2$ for user authentication Chooses ' r_3 ' and computes: $SDid_j = ((id_i \oplus Sid_j) \oplus h(SMK_j r_3 n),$ $N_j = n \oplus h(SMK_j r_3),$ $M_3 = h(ID_i Sid_j Z_i^* n SMK_j r_3),$ $Z_j = Z_i^* \oplus h(r_3 n SMK_j Sid_j ID_i),$ $R_j = r_2 \oplus h(SMK_j id_i r_3).$ $\{SDid_j, N_j, Z_j, R_j, M_3, r_3, h(\cdot)\}$								
	$n = N_j \oplus h(SMK_j r_3),$ $(id_i \oplus Sid_j) = SDid_j \oplus h(SMK_j r_3 n),$ $id_i = Sid_j \oplus (id_i \oplus Sid_j),$ $Z_i = Z_j \oplus h(r_3 n SMK_j Sid_j id_i),$ $M_3^* = h(id_i Sid_j Z_i^* n SMK_j r_3)$ Verifies $M_3^* =? M_3$ for authentication Retrieves $r_2 = R_j \oplus h(SMK_j id_i r_3)$ and chooses ' r_4 ' $SK_{ij} = h(id_i r_2 Sid_j h(Z_i^*) r_3 r_4)$ Computes $M_4 = h(SK_{ij} r_4 id_i Sid_j),$ $S_{j1} = r_4 \oplus h(id_i r_2 Z_i),$ $S_{j2} = r_3 \oplus h(id_i Z_i r_4)$ Login reply msg $\{M_4, S_{j1}, S_{j2}\}$							
		$r_4^* = S_{j1} \oplus h(id_i r_2 Z_i), r_3^* = S_{j2} \oplus h(id_i Z_i r_4),$ $SK_{ij}^* = h(id_i r_2 Sid_j h(Z_i) r_3^* r_4^*)$ $M_4^* = h(SK_{ij} r_4^* id_i Sid_j),$ Verifies $M_4^* =? M_4.$						
Now user, sensor, BS agree on common session key $SK_{ij} = h(id_i r_2 Sid_j h(Z_i^*) r_3 r_4)$								

Fig. 2. Graphical view of the proposed procedure.

H. Resistance Against the Off-Line Dictionary Password Guessing Attack

In this kind of attack, the foe somehow captures the messages exchanged among a client, the BS, and a sensor node, namely, $\{Did_i, Sid_j, M_1, M_2, r_1, e_i\}, \{SDid_j, M_3, N_j, R_j, Z_j, r_3, h(\cdot)\}, \{M_4, S_{j1}, S_{j2}\}$ in our procedure, and then attempts to guess the client's id_i and PW_i offline from the recorded messages. However, the attacker cannot come to the right

id_i , and PW_i even if he/she has the messages. Therefore, the proposed procedure is protected against the offline dictionary password guessing attack.

I. Achieving Mutual Authentication

In our new procedure, the three conveying parties, namely, the client, the BS, and the sensor node, mutually authenticate each other. All the three parties give their arbitrary

numbers r_2 , r_3 , and r_4 for the derivation of the session key $SK_{ij} = h(id_i || r_2 || Sid_j || h(Z_i^*) || r_3 || r_4)$. The BS authenticates the client C_i using such verification data as $M_2^* = h(id_i || Sid_j || r_1 || r_2 || e_i || Z_i^*)$; the sensor node authenticates the BS using $M_3^* = h(id_i || Sid_j || Z_i^* || n || SMK_j || r_3)$; and the client C_i authenticates the sensor node and the BS using $M_4^* = h(SK_{ij} || r_4 || id_i || Sid_j)$. Obviously, the proposed procedure satisfies the requirement of mutual authentication.

J. Resistance Against Attacks Launched by a Malicious Client

A legitimate but malicious client with his/her own SC can collect data such as $\{c_i, f_i, E_i, N_i, \tau_i, h(\cdot)\}$, where $c_i = Z_i \oplus RPW_i$; $f_i = h(Rid_i || n || RPW_i)$; $E_i = e_i \oplus h(id_i || \sigma_i || PW_i)$; $N_i = n \oplus h(id_i || \sigma_i || PW_i)$ and $Z_i = h(Rid_i || X_s || n)$, from the SC memory, but in our procedure those data would not help the malicious client guess any other valid client's password (PW_i) or the BS's secret key (X_s), since it would only be in vain for the malicious client to substitute his/her own values (id_i, PW_i, σ_i, n) in the above equations trying to guess another valid client's password or BS's secret key.

K. Perfect Forward Secrecy of the Session Key

In the authentication protocol, the session key's perfect forward security means that a session key extracted from a collection of long-term keys will not be compromised even if the long-term keys are compromised in the future. In our proposed procedure, $SK_{ij} = h(id_i || r_2 || Sid_j || h(Z_i^*) || r_3 || r_4)$ is a shared session key between the sensor node, user, and the BS. Even if a foe gets the secret key of the X_s gateway node, BS, or the password PW_i of the user, he/she is still not able to compute old session keys because the temporary numbers, r_2, r_3 , and r_4 are randomly chosen and independent between protocol executions. Therefore the foe is unable to calculate the session key. Consequently, the proposed procedure maintains the perfect forward secrecy of the session key.

VII. FORMAL SECURITY INVESTIGATION OF IMPROVED PROCEDURE USING RANDOM ORACLE MODEL

Reveal: This is an oracle, which genuinely yields the info string x from the comparing hash value y where $y = h(x)$.

Theorem 1: Under the supposition that a one-way hash function $h(\cdot)$ works in a very similar way to an oracle, our procedure is secure in resisting an attacker trying to capture id_i of a legitimate client C_i and the secret key X_S of the BS, even if the client C_i 's smart card SC is lost/stolen and the messages exchanged for communication are intercepted.

Proof: In this proof, we accept that the foe E has the stolen/lost smart card SC of C_i and can excerpt all the client specific data from its memory utilizing the power analysis attack [2]–[6] and can determine ID_i, PW_i, σ_i of a legal client C_i as well as the secret information X_S of the BS. For this purpose, we expect that E approaches the oracle *Reveal*. E at that point utilizes the *Reveal* oracle for running an exploratory algorithm, namely, $\text{EXP1}^{\text{HASH}}_{E, \text{AUS}}$, which is portrayed in Fig. 1. The success probability for $\text{EXP1}^{\text{HASH}}_{E, \text{AUS}}$ can be defined as $\text{Succ1}^{\text{HASH}}_{E, \text{AUS}} = |\text{Pr}[\text{EXP1}^{\text{HASH}}_{E, \text{AUS}} = 1] - 1|$, where $\text{Pr}[\text{Ev}]$ is the possibility of occurring of an event "Ev." The advantage function for this examination becomes $\text{Adv1}(et_1, qr) = \max_E \{\text{Succ1}^{\text{HASH}}_{E, \text{AUS}}\}$, where the most extreme scenario is assumed control over all "E"

```

% OFMC
% Version of 2006/02/13
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
/home/avispa/web-interface-computation/
./tempdir/codefilewa4NHD.if
GOAL
as_input
BACKEND
OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime: 45.53 s
visitedNodes: 5423 nodes
depth: 7 plies

```

Fig. 3. Result of the formal security verification of our procedure using OFMC.

TABLE II
FUNCTIONALITY AND SECURITY COMPARISON AMONG SIMILAR PROCEDURES

SP\ Procedures	[20]	[21]	[22]	[41]	[23]	[15]	Proposed Procedure
SP_1	Yes	Yes	No	No	Yes	Yes	Yes
SP_2	Yes	No	Yes	Yes	No	Yes	Yes
SP_3	No	No	No	No	No	Yes	Yes
SP_4	Yes	No	No	Yes	No	Yes	Yes
SP_5	No	Yes	Yes	Yes	Yes	No	Yes
SP_6	No	No	No	Yes	No	No	Yes
SP_7	Yes	Yes	Yes	Yes	Yes	No	Yes
SP_8	No	No	No	No	No	Yes	Yes
SP_9	No	No	No	No	No	No	Yes

SP₁: Password change; SP₂: Mutual authentication; SP₃: Biometric update; SP₄: DoS attack resistance; SP₅: Client anonymity; SP₆: Replay attack resistance; SP₇: Session specific temporal information attack resistance; SP₈: Proof by formal security analysis; SP₉: Perfect forward secrecy.

with execution time et_1 and the number of queries qr made to the *Reveal* oracle.

We assure that our procedure is secure against the attacker E for deriving PW_i, σ_i and X_S , if $\text{Adv1}(et_1, qr) \leq \alpha$, for a small $\alpha > 0$. As indicated by the analysis given in Algorithm 1, unmistakably if E can transform the hash function $h(\cdot)$, at that point he/she determines PW_i, σ_i , and X_S , E wins the game. It is computationally infeasible to invert $h(\cdot)$, i.e., $\text{Adv}^{\text{HASH}}_E(t_1) \leq \alpha_1$, for any adequately small $\alpha_1 > 0$ in polynomial time. In this way, we have $\text{Adv1}(et_1, qr) \leq \alpha$, since $\text{Adv1}(et_1, qr)$ depends on the advantage $\text{Adv}^{\text{HASH}}_E(t_1)$. Subsequently, our procedure is secure against a foe "E" for capturing PW_i, σ_i , and X_S regardless of the possibility that the client C_i 's smart card SC is lost/stolen.

In addition to the above formal security examination, we also dissected the security strengths of our procedure by utilizing another broadly acknowledged verification tool, namely, AVISPA [3]–[6]. AVISPA is regarded as a push button tool for automated validation of authentication procedures and applications which are vulnerable to Internet security. AVISPA offers a standardized language that is flexible and descriptive to define authentication procedures and their security properties, and incorporates various back ends that incorporate a number of state-of-the-art automated analytics [39]. The first back-end, known as the on-the-fly model-checker (OFMC), uses many symbolic techniques to explore the state space in a demand-driven manner [40]. The simulation result for the formal security confirmation

TABLE III
COMPARISON OF COMPUTATION COST AMONG VARIOUS PROCEDURES

Procedures\Stages	Registration	Login+ Authentication	Total	Estimated time (ms)
[20]	$7 t_H + 3 t_{XOR}$	$14 t_H + 9 t_{XOR}$	$21 t_H + 12 t_{XOR}$	≈ 0.048
[21]	$6 t_H + 2 t_{XOR}$	$14 t_H + 12 t_{XOR}$	$20 t_H + 14 t_{XOR}$	≈ 0.046
[22]	$5 t_H$	$15 t_H + t_{XOR}$	$20 t_H + t_{XOR}$	≈ 0.046
[41]	$6 t_H + 4 t_{XOR}$	$15 t_H + 10 t_{XOR}$	$21 t_H + 14 t_{XOR}$	≈ 0.048
[23]	$4 t_H + t_{XOR}$	$14 t_H + 5 t_{XOR}$	$20 t_H + 6 t_{XOR}$	≈ 0.048
[15]	$5 t_H + t_{Fe} + 5 t_{XOR}$	$13 t_H + t_{Fe} + t_E + 11 t_{XOR}$	$18 t_H + t_D + t_{Fe} + t_E + 16 t_{XOR}$	≈ 0.056
Proposed Procedure	$5 t_H + t_{Fe} + 4 t_{XOR}$	$11 t_H + t_{Fe} + 9 t_{XOR}$	$15 t_H + t_{Fe} + t_E + 13 t_{XOR}$	≈ 0.043

TABLE IV
COMPARISON OF COMMUNICATION COSTS BETWEEN OUR PROCEDURE AND OTHER PROCEDURES

Procedures	Communication cost
[20]	5 Messages (944 bits)
[21]	3 Messages (736 bits)
[22]	4 Messages (944 bits)
[41]	5 Messages (1,232 bits)
[23]	3 Messages (704 bits)
[15]	3 Messages (736 bits)
Proposed Procedure	3 Messages (736 bits)

Algorithm 1: EXP1^{HASH}_{E,AUS}.

1. Extract the values or data stored in the C_i S.C, i.e., $\{c_i, f_i, E_i, N_i, \tau_i, h(\cdot)\}$ using various techniques as discussed in Watro *et al.*[16].
2. Call *Reveal* (f_i) to retrieve Rid_i^* , n^* , RPW_i^* .
3. Call *Reveal* (RPW_i^*) to retrieve id_i^* , σ_i^* , PW_i^* , n^{**} .
4. if $n^* = n^{**}$
5. Accept PW_i^* , σ_i^* as the correct password and biometrics of U_i .
6. Call *Reveal* (M_2) to retrieve id_i^{**} , Sid_j^* , r_1^* , r_2^* , e_i^* , Z_i^*
7. Call *Reveal* (Z_i^*) to retrieve Rid_i^{**} , X_S , n^{***} .
8. If $Rid_i^* = Rid_i^{**}$, Accept id_i^* as the real identity and X_S as the *B.S* secret key.
Return 1 (success)
Else Return 0 (failure)
End if
Else, Return 0 (failure)
End if.

of our procedure utilizing OFMC is shown in Fig. 3, and the results confirm that the proposed procedure is attack-resistant and secure.

VIII. PERFORMANCE ANALYSIS

To have a clear picture of how well our procedure can perform, we have compared it with some related procedures such as [15], [20]–[23], [41] in terms of some essential functionalities and security features, and the comparison results are shown in Table II. It is obvious that our new procedure is superior over the other procedures in that it provides a password change mechanism, allows users to update biometric data, supports mutual authentication as well as client anonymity, and is strong against the replay attack, the DoS attack, and the session specific temporal information attack. Besides, the security protection

Algorithm 2: EXP2^{HASH}_{E,AUS}.

1. Intercept the login request during the login stage, i.e., $\{Did_i, Sid_j, M_1, M_2, e_i, r_1\}$.
2. Extract the values or data stored in C_i 's S.C i.e $\{c_i, f_i, E_i, N_i, \tau_i, h(\cdot)\}$ using various techniques as discussed in Watro *et al.* [16], where $E_i = e_i \oplus h(id_i || \sigma_i || PW_i)$, $N_i = n \oplus h(id_i \oplus \sigma_i \oplus PW_i)$ and
3. Call *Reveal* (E_i) to retrieve e_i^* , id_i^* , σ_i^* , PW_i^* .
4. Call *Reveal* (N_i) to retrieve n^* , id_i^{**} , σ_i^{**} , PW_i^{**}
5. Call *Reveal* (M_2) to retrieve id_i^{**} , Sid_j^* , r_1^* , r_2^* , e_i^* , Z_i^*
6. If ($e_i^* = e_i^{**}$)
7. Compute $Did_i^* = id_i^* \oplus h(e_i^* \oplus n^*) \oplus r_1^*$
8. If $Did_i^* = Did_i$
9. Accept id_i^* as the correct identity id_i of C_i .
10. Call *Reveal* (M_2) to retrieve id_i^{**} , Sid_j^* , r_1^* , r_2^* , e_i^{**} , Z_i^*
11. Call *Reveal* (Z_i^*) to retrieve Rid_i^{**} , X_S , n^{***} .
12. Compute $Rid_i^{***} = h(id_i^* || n^{***})$
13. If $Rid_i^{***} = Rid_i^{**}$
14. Accept id_i^* as the correct identity of the client and X_S as the *B.S* secret key.
Return 1 (success)
Else Return 0 (failure)
End if
Else, Return 0 (failure)
End if.
Else, Return 0 (failure)
End if.

of our new procedure has been proven by a formal security examination and a formal verification.

In addition, we have compared our new procedure with those same related procedures in terms of computation cost during the registration stage, login stage, and authentication stage. The comparison results are shown in Table III, where t_H denotes time for a hash computation [42], t_E denotes time for symmetric encryption (AES encryption [43]), t_D denotes time for symmetric decryption (AES decryption [43]), and t_{Fe} denotes time for implementing a fuzzy extractor function [Rep (\cdot) and Gen(\cdot)].

Kilinc and Yanik [44] have recently introduced cryptographic primitives utilizing the PBC library (version 0.5.12) under an Ubuntu 12.04.132-bit operating system with a 2.2 GHz CPU and 2.0 GB RAM. It should be noted that other researchers used the recorded findings of [44] to adopt a homogeneous method to calculate the total computation cost of their procedures [45].

TABLE V
COMPARISON OF SENSOR NODE'S ENERGY STORAGE COST BETWEEN OUR PROCEDURE AND OTHER PROCEDURES

Procedures	Sensor node's energy storage cost
[20]	One hash operation for random nonce validation + one hash operation for session key generation + response to the user's query
[21]	Timestamp validation + one hash operation for parameter generation + response to the user's query
[22]	Timestamp validation + one hash operation for parameter generation + response to the user's query
[41]	One hash operation for random nonce validation + one hash operation for session key generation + response to the user's query
[23]	Timestamp validation + one hash operation for parameter generation + response to the user's query
[15]	Timestamp validation + One symmetric-key decryption and encryption +two hash operations for session key generation and validation + response to the user's query
Proposed Procedure	One symmetric-key encryption + timestamp validation +two hash operations for session key generation and validation + response to the user's query

Kilinc and Yanik [44] stated that the average execution times for one symmetrical encryption/decryption operation, one hash operation, one modular multiplication, and one modular exponentiation were 0.0046, 0.0023, and 0.001855 ms, respectively. Time implementing a fuzzy extractor function is similar as one symmetric encryption/decryption operation [46]. Note that during the registration stage our procedure requires a total computation cost of $(5t_H + t_{Fe} + 4t_{XOR})$, which includes all the computations on the client side, on the BS side, and on the sensor node side. During the login stage and in the authentication procedure, our procedure requires a total computation cost of $(11t_H + t_{Fe} + 9t_{XOR})$. Due to the high efficiency performance of the fuzzy extractor, burdens on both the client's side and the BS's side are effectively reduced. In addition, the time of execution of the XOR process is considered negligible. The estimated time needed to implement the proposed authentication procedure is therefore 0.043 ms.

Table III summarizes the computing costs and average time for implementing the associated authentication and key agreement procedures [15], [20]–[23], [41]. On the other hand, our employment of one-way hash function $h(\cdot)$ and symmetric decryption/encryption also lowers the computation cost on the sensor node's side. As a result, our procedure is very suitable for networks equipped with resource-constrained sensor nodes.

The communication costs for our presented procedure and the other procedures are shown in Table IV in terms of the number of exchanged messages and bits for an effective user authentication. We used the number of bits needed between our procedure and other procedures for the following fields to measure the communication costs. The identifiers of BS (GW-node), sensor nodes, and user are every 16 bits. The random nonce and time-stamp fields are 32 bits. The encryption and decryption processes use the AES with 128 bits. When we use SHA-1 as a one-way hash function the digest message is 160 bits. It is clear from Table IV that an effective user authentication process in our presented procedure needs 736 bits, while procedures [20]–[23], and [41] require 944, 736, 944, 704, and 1232 bits, respectively.

Lastly, in Table V, we compared the energy costs needed for a sensor node between our procedure and other procedures.

Remember that the energy cost of a sensor node is attributed largely to both the computation and communication costs involved in the procedures. For the procedure reported in [15], the battery consumption for a sensor node is correlated with time-stamp validation, one symmetric decryption, and two hash functions to generate parameters and respond to user query. In the procedure in [20], battery consumption for a sensor

node is induced by validation of timestamps, one hash function for parameter generation, another hash function for parameter verification, and then the user's question is answered and the GW-node's response is awaited. The procedure in [21], a sensor node consumes batteries due to validation of timestamps, one hash function for generation of parameters and an answer to the user's query. Procedure in [22] includes battery consumption for a sensor node due to time-stamp validation, one hash function for the generation of parameters, and the user query response. In the procedure in [23], a sensor node consumes battery for the generation of parameters due to time-stamp validation and one hash process, and finally to respond to user question. Procedure in [41] needs battery consumption for a sensor node due to one hash function for random-nonce validation, another hash function for session key generation, and then the user query response. Finally, in our procedure, a sensor node consumes battery due to one symmetric encryption, timestamp validation, two hash operations for session key generation and validation, and response to the user's query. Due to efficient hash and symmetric-key operations, a sensor node's energy cost in our procedure is comparable with that for the other procedures.

IX. CONCLUSION

In this article, we have enhanced the 3-FA procedure for large-scale DWSN that was projected by Das. We have demonstrated that besides having a time synchronization issue, Das's procedure also fails to achieve client anonymity and is vulnerable to the session specific temporary information attack, the replay attack, etc. To solve these problems, we have constructed a more robust authentication procedure that is secure against all of the major possible attacks. In addition, instead of encryption and decryption, our new procedure uses only lightweight computation-based procedure such as hash and Ex-OR operations, which makes it especially suitable for network environments equipped with low computation capacity devices like sensors. Hence, our procedure is extremely applicable in real life WSN systems. Our security and efficiency evaluations have demonstrated that our devised procedure has better performance than other competing similar schemes.

REFERENCES

- [1] M. S. Obaidat and S. Misra, *Principles of Wireless Sensor Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
- [2] X. Li, J. Niu, S. Kumari, J. Liao, and W. Liang, "An enhancement of a smart card authentication protocol for multi-server architecture," *Wireless Pers. Commun.*, vol. 80, no. 1, pp. 175–192, 2015.

- [3] Q. Jiang, J. Ma, X. Lu, and Y. Tian, "An efficient two-factor client authentication protocol with unlinkability for wireless sensor networks," *Peer-to-Peer Netw. Appl.*, vol. 8, no. 6, pp. 1070–1081, 2015.
- [4] D. He, N. Kumar, J. Chen, C. C. Lee, N. Chilamkurti, and S. S. Ye, "Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks," *J. Multimedia Syst.*, vol. 21, no. 1, pp. 49–60, 2015.
- [5] S. Khan and R. Khan, "A secure authentication and key management protocol for wireless sensor networks," in *Proc. 5th Int. Conf. Sensor Syst. Softw. Coventry, U.K.*, 2015, vol. 143, pp. 51–60.
- [6] O. R. M. Boudiaa, S. M. Senoucib, and M. Fehama, "A novel secure aggregation protocol for wireless sensor networks using stateful public key cryptography," *Ad Hoc Netw.*, vol. 32, pp. 98–113, 2015.
- [7] U. S. Kumaran and P. Ilango, "Secure authentication and integrity techniques for randomized secured routing in WSN," *Wireless Netw.*, vol. 21, no. 2, pp. 443–451, 2015.
- [8] X. Liu and Q. Zhao, "Cluster key protocol based on bilinear pairing for wireless sensor networks," in *Proc. 4th Int. Conf. Comput. Eng. Netw.*, 2015, pp. 299–304.
- [9] M. S. Obaidat and D. T. Macchiarolo, "A multilayer neural network system for computer access security," *IEEE Trans. Syst., Man Cybern. B*, vol. 24, no. 5, pp. 806–830, May 1995.
- [10] M. S. Obaidat and B. Sadoun, "Verification of computer users using keystroke dynamics," *IEEE Trans. Syst., Man Cybern. B*, vol. 27, no. 2, pp. 261–269, Apr. 1997.
- [11] C. T. Li and M. S. Hwang, "An efficient biometric-based remote authentication protocol using smart cards," *J. Netw. Comput. Appl.*, vol. 33, no. 1, pp. 1–5, 2010.
- [12] X. Li, J. W. Niu, J. Ma, W. D. Wang, and C. L. Liu, "Cryptanalysis and improvement of a biometrics-based remote user authentication protocol using smart cards," *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 73–79, 2011.
- [13] Y. Choi, J. Nam, Y. Lee, S. Jung, and D. Won, "Cryptanalysis of advanced biometric based client authentication protocol for wireless sensor networks," in *Computer Science and Its Applications*, Berlin, Heidelberg: Springer, vol. 330, 2015, pp. 1367–1375.
- [14] M. S. Obaidat, I. Traoc, and I. Woungnag, *Biometric-Based Physical and Cyber Security Systems*. Vienna, Austria: Springer, 2018.
- [15] A. K. Das, "A secure and efficient client anonymity-preserving three-factor authentication protocol for large-scale distributed wireless sensor networks," *Wireless Personal Commun.*, vol. 82, no. 3, pp. 1377–1404, 2015.
- [16] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus, "TinyPK: Securing sensor networks with public key technology," in *Proc. 2nd ACM Workshop Secur. Ad Hoc Sensor Netw.*, 2004, pp. 59–64.
- [17] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.
- [18] M. L. Das, "Two-Factor client authentication in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, pp. 1086–1090, Mar. 2009.
- [19] K. Wong, Y. Zheng, J. Cao, and S. Wang, "A dynamic client authentication protocol for wireless sensor networks," in *Proc. IEEE Int. Conf. Sensor Netw., Ubiquitous, Trustworthy Comput., IEEE Comput. Soc.*, 2006, pp. 244–251.
- [20] B. Vaidya, D. Makrakis, and H. T. Mouftah, "Improved two-factor client authentication in wireless sensor networks," in *Proc. 6th 2010 IEEE 6th Int. Conf. Wireless Mobile Comput., Netw. Commun.*, 2010, pp. 600–606.
- [21] D. He, Y. Gao, S. Chan, C. Chen, and J. Bu, "An enhanced two-factor client authentication protocol in wireless sensor networks," *Ad Hoc Sensor Wireless Netw.*, vol. 10, no. 4, pp. 361–371, 2010.
- [22] T. H. Chen and W. K. Shih, "A robust mutual authentication protocol for wireless sensor networks," *ETRI J.*, vol. 32, no. 5, pp. 704–710, 2010.
- [23] J. Yuan, C. Jiang, and C. Jiang, "A biometric-based client authentication for wireless sensor networks," *Wuhan Univ. J. Natural Sci.*, vol. 15, no. 3, pp. 272–276, 2010.
- [24] M. K. Khan, S. K. Kim, and K. Alghathbara, "Cryptanalysis and security enhancement of a more efficient & secure dynamic ID-based remote client authentication protocol," *Comput. Commun.*, vol. 34, no. 3, pp. 305–309, 2011.
- [25] C. L. Chen, C. C. Lee, and C. Y. Hsu, "Mobile device integration of a fingerprint biometric remote authentication protocol," *Int. J. Commun. Syst.*, vol. 25, no. 5, pp. 585–597, 2012.
- [26] T. T. Truong, M. T. Tran, and A. D. Duong, "Robust mobile device integration of a fingerprint biometric remote authentication scheme," in *Proc. IEEE 26th Int. Conf. Adv. Inf. Netw. Appl.*, 2012, pp. 678–685.
- [27] M. K. Khan, S. Kumari, and M. K. Gupta, "More efficient key-hash based fingerprint remote authentication protocol using mobile device," *Computing*, vol. 96, no. 9, pp. 793–816, 2014.
- [28] R. Amin and G. P. Biswas, "A secure light weight protocol for client authentication and key agreement in multi-gateway based wireless sensor networks," *Ad Hoc Netw.*, vol. 36, pp. 58–80, 2016.
- [29] A. Ostad-Sharif, H. Arshad, M. Nikooghadam, and D. Abbasinezhad-Mood, "Three party secure data transmission in IoT networks through design of a lightweight authenticated key agreement scheme," *Future Gener. Comput. Syst.*, vol. 100, pp. 882–892, 2019.
- [30] D. Abbasinezhad-Mood and M. Nikooghadam, "Design of an enhanced subtree-based authentication scheme for smart grid and its performance analysis on an ARM Cortex-M3 microcontroller," *J. Inf. Secur. Appl.*, vol. 40, pp. 9–19, 2018.
- [31] D. Abbasinezhad-Mood and M. Nikooghadam, "Design and hardware implementation of a security-enhanced elliptic curve cryptography based lightweight authentication scheme for smart grid communications," *Future Gener. Comput. Syst.*, vol. 84, pp. 47–57, 2018.
- [32] D. Abbasinezhad-Mood and M. Nikooghadam, "Efficient anonymous password-authenticated key exchange protocol to read isolated smart meters by utilization of extended chebyshev chaotic maps," *IEEE Trans. Inf. Inform.*, vol. 14, no. 11, pp. 4815–4828, Nov. 2018.
- [33] W. Liu, J. Liu, Q. Wu, B. Qin, D. Naccache, and H. Ferradi, "Efficient subtree-based encryption for fuzzy-entity data sharing," *Soft Comput.*, vol. 22, pp. 7961–7976, 2018.
- [34] R. Canetti and H. Krawczyk, "Analysis of key exchange protocols and their use for building secure channels," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, London, U.K., 2001, pp. 453–474.
- [35] Z. Cheng, L. Vasiu, R. Comley, and M. Nistazakis, "On the indistinguishability based security model of key agreement protocols simple cases," *Cryptol. ePrint Archive*, 2005.
- [36] C. M. Swanson, "Security in Key Agreement: Two Party Certificateless Protocols," Master's thesis, Univ. Waterloo, Waterloo, ON, Canada, 2008.
- [37] T. Mandt and C. Tan, "Certificateless authenticated two party key agreement protocols," in *Proc. Annu. Asian Comput. Sci. Conf.*, vol. 4435, pp. 37–44, 2008.
- [38] M. Hou, H. Jiang, G. Shanqing, and Q. Xu, "Cryptanalysis of identity based authenticated key agreement protocols from pairings," *J. Netw.*, vol. 5, no. 7, pp. 826–855, 2010.
- [39] AVISPA. Automated Validation of Internet Security Protocols and Applications. Accessed: Jan. 2013. [Online]. Available: <http://www.avispa-project.org/>
- [40] D. Basin, S. Modersheim, and L. Vigano, "OFMC: A symbolic model checker for security protocols," *Int. J. Inf. Secur.*, vol. 4, no. 3, pp. 181–208, 2005.
- [41] R. Fan, L. D. Ping, J. Q. Fu, and X. Z. Pan, "A secure and efficient client authentication protocol for two-tier wireless sensor networks," in *Proc. 2nd Pacific-Asia Conf. Circuits, Commun. System*, 2010, pp. 425–428.
- [42] Secure Hash Standard. FIPS PUB 180–1, National Institute of Standards and Technology (NIST), U.S., 1995.
- [43] W. Stallings, *Cryptography and Network Security: Principles and Practices*, 3rd ed. Gaithersburg, MD, USA: Pearson Education India, 2003.
- [44] H. Kilinc and T. Yanik, "A survey of SIP authentication and key agreement schemes," *IEEE Commun. Surv. Tut.*, vol. 16, no. 2, pp. 1005–1023, Second Quarter 2014.
- [45] M. Nikooghadam, R. Jahantigh, and H. Arshad, "A lightweight authentication and key agreement protocol preserving user anonymity," *Multimed Tools Appl.*, vol. 76, pp. 13401–13423, 2017.
- [46] L. Zhang and Y. Nie, "A secure authentication scheme based on fuzzy extractor," *Comput. Modelling New Technol.*, vol. 18, no. 12C, pp. 46–55, 2014.