A Secure and Lightweight Three-Factor Remote User Authentication Protocol for Future IoT Applications

Bahaa Hussein Taher, Huiyu Liu, Firas Abedi, Hongwei Lu, Ali A. Yassin, and Alzahraa J. Mohammed

1School of Computer Science, Huazhong University of Science and Technology, Wuhan, 400037 Hubei, China
2Department of Mathematics, College of Science, University of Basrah, 61004, Iraq
3Medical Instrumentation Engineering Techniques Department, Al-Mustaqbal University College, 51001 Hillah, Babil, Iraq
4Department of Computer Science, Education College of Pure Science, University of Basrah, 61004, Iraq

Correspondence should be addressed to Huiyu Liu; liuhuiyu@hust.edu.cn

Received 22 August 2020; Revised 24 October 2020; Accepted 28 November 2020; Published 28 April 2021

Academic Editor: ahmed alkhayyat

Copyright © 2021 Bahaa Hussein Taher et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the booming integration of IoT technology in our daily life applications such as smart industrial, smart city, smart home, smart grid, and healthcare, it is essential to ensure the security and privacy challenges of these systems. Furthermore, time-critical IoT applications in healthcare require access from external parties (users) to their real-time private information via wireless communication devices. Therefore, challenges such as user authentication must be addressed in IoT wireless sensor networks (WSNs). In this paper, we propose a secure and lightweight three-factor (3FA) user authentication protocol based on feature extraction of user biometrics for future IoT WSN applications. The proposed protocol is based on the hash and XOR operations, including (i) a 3-factor authentication (i.e., smart device, biometrics, and user password); (ii) shared session key; (iii) mutual authentication; and (iv) key freshness. We demonstrate the proposed protocol’s security using the widely accepted Burrows–Abadi–Needham (BAN) logic, Automated Validation of Internet Security Protocols and Applications (AVISPA) simulation tool, and the informal security analysis that demonstrates its other features. In addition, our simulations prove that the proposed protocol is superior to the existing related authentication protocols, in terms of security and functionality features, along with communication and computation overheads. Moreover, the proposed protocol can be utilized efficiently in most of IoT’s WSN applications, such as wireless healthcare sensor networks.

1. Introduction

The IoT has been a trend in the last few years, and it is expected to be so in the future [1]. In an IoT system, information is being sensed/collected by IoT sensing devices such as embedded systems, Radio Frequency Identification (RFID), wearable devices, and low powered IEEE 802.15.4 devices before being sent to another intermediary device/node (e.g., edge or fog computing node), IoT device, or to the cloud, via the Internet. In IoT, many devices can interact with each other over the Internet. A lot of IoT applications, already, have been deployed such as healthcare systems, smart cities, smart industrial, transportation systems, and smart homes [2, 3]. In these IoT applications, WSNs are most necessary and important [4]. The use of WSNs has greatly increased in providing services to activities and monitoring environments due to its low costs, ease of deployment, a wide range of applications, and flexibility [5]. Therefore, security and privacy are a significant challenge in any consumer technology deployment [6]. For example, let us highlight on an IoT healthcare application [7] as shown in Figure 1. In this scenario, the quality of healthcare service can be enhanced by allowing a medical practitioner to direct access to data that have sensed by the medical sensor nodes deployed in his patient’s body. This information can involve current vital reading such as blood pressure, cholesterol, C-reactive protein, and blood sugar level. Accordingly, based on this private and secret current information, a decision can be taken...
regarding the patient’s health condition to provide necessary remedial actions.

In IoT, the sensor node/devices in WSN face a significant security challenge. Usually, those sensor nodes are deployed in places that are easy for people to touch. Nowadays, one of the most possible critical security attacks that easily happen is a node captured attack where the authentication information that is inside the sensor node is revealed by physical crack [8]. Furthermore, new remote user authentication that is also possible could be vulnerable to this attack because the malicious attacker possibly obtained all sensitive authentication information through this attack. Thus, the security issues in IoT WSN applications are significant and catch more attention. To satisfy this goal, we proposed a secure and lightweight remote user authentication and key agreement protocol to operate in an IoT WSN application environment.

1.1. Motivation. The IoT WSN has opened up many opportunities in various walks of life and particularly in healthcare, shipping, warehousing, and logistics, which have facilitated processes for consumers and businesses. This wide-ranging and rapid development has led to the emergence of great challenges that require the design of high-security protocols for IoT applications in order to preserve the sensitive information of users. Security is now the primary challenge facing the IoT WSN environment. In an IoT WSNs, remote users can access data from IoT sensor nodes via the Internet. Researchers have developed effective mechanisms to integrate wireless networks into the Internet of things environments [9, 10]. Sensor nodes are inherently resource-constrained devices in terms of limited processing capability, constrained communication bandwidth, and very low storage capacity due to the physical size and limited energy [11]. Therefore, designing a secure and efficient remote user authentication protocol for IoT WSN environments is a non-trivial challenge. In IoT environments, the security efficiency of remote user authentication is an important issue for transmitting information securely [12–14]. In addition, energy consumption and computational and communication efficiency are crucial due to the WSN resources energy limitation. Also, due to constrained sensors, adding resourceful gateway nodes can support the sensors, can provide quick on-demand delivery of information, and take care of most of the processing. The authentication of users or devices is a critical issue that must be considered in the context of IoT security. Most of the traditional authentication protocols are based on a password, a smart card, or both. These protocols are ineffective at present since attackers have modified the methodology of their attacks on IoT devices. The need for biometric-based approaches that are difficult to reproduce, such as those involving fingerprints, iris scanning, and facial patterns, has emerged as an additional factor that can enhance the security of Internet applications.

1.2. Attack Model. In our proposed protocol, we follow the widely accepted and more realistic Dolev-Yao threat (DY) model [15]. In this model, the communication between two entities is accomplished over a public (open) channel. Also, an adversary MA will have full control over the communication channel. Therefore, MA can alter, eavesdrop, insert, and delete forgery messages that are being transmitted during communication. In addition, it is assumed that MA can physically capture one or more IoT sensing nodes in IoT
and can steal all sensitive information stored in the captured sensing nodes which utilize the strength analysis attacks.

1.3. Our Contribution. The main contributions of our proposal are as follows:

1. We proposed a lightweight and secure remote user authentication protocol based on feature extraction of the user fingerprint and one-way hash function for IoT WSN applications which is suitable to use in wireless healthcare application. The proposed protocol is three factors: user password, smartphone, and biometrics to achieve our goal. We used biometrics to increase the security of the protocol due to difficulty to forge or steal or forget biometrics.

2. Level 3 feature extraction is done to overcome the problem of noise in fingerprint images in existing authentication schemes.

3. We prove our protocol secure using informal and formal security analysis through BAN logic and random oracle model.

4. We simulate the proposed protocol using the popular and widely accepted tool called AVISPA and demonstrate that the protocol is perfectly secure against active and passive attacks.

5. Comparative evaluation of our protocol with related protocols in terms of communication and computational overheads was performed.

2. Related Work

The general security requirements needed to secure an IoT WSN environment are authentication, integrity, confidentiality, availability, nonrepudiation, authorization, freshness, forward, and backward secrecy. Therefore, a remote user authentication scheme designed for an IoT WSN environment should be designed in a way that ensures it withstand many attacks such as man-in-the-middle, online/offline guessing, replay, privileged insider, stolen/lost smart card, password change, and sensing device capture. Also, the designed scheme should reduce computation and communication costs and include the password/biometric update phase. Presume a scenario for a medical practitioner wandering the medical IoT environment. In such an assumption, we need to preserve certain information about this user such as achieving anonymity preservation to prevent other parties (users) from revealing the patient’s critical privacy information while he/she joins the system sessions. By way of explanation, user anonymity is one of important key features in the user authentication protocol [16]. Also, the untraceability feature is important in the IoT WSN applications to prevent an attacker from tracing a user during a session [17]. WSNs have become an important and necessary network infrastructure after modernization, and they can be generally used in many modern fields such as health monitoring, environments, and smart homes [18–21]. To gratify the security requirements of the IoT WSN environment, numerous user authentication protocols have been proposed. Shi and Gong [22] proposed a new user-authentication scheme using ECC for the WSNs. Unfortunately, the storage and computational overhead are relatively high so it is not applicable for healthcare application systems [23–25].

Usually, the real-time users adopt to use easy-to-memorization parameters, such as secret keys and identities, for their convenience, as explain in [23–25]; hence, user anonymity is not provided. For the enhancement security of IoT WSN, studies in [26–30] presented lightweight remote user authentication schemes. Nonetheless, these contributions have need of improvements to resist attacks while persevering optimum communication and computation performance. In 2016, Arasteh et al. [31] proposed an authentication scheme for an IoT network that aimed to overcome the weaknesses of a scheme designed by Amin et al. [32]. In 2017, Dhillon and Kalra’s [33] proposed a lightweight 3FA scheme using a user password, biometric, and a mobile device. They pointed out that their scheme is secure against well-known attacks such as a denial of service, impersonation, offline password guessing, and stolen mobile device attacks. However, their scheme is still insecure against the mentioned attacks and does not afford a session key agreement. In the same year, Li et al. [34] and Zhang et al. [35] presented their authentication schemes with key agreement. They showed that their scheme was lightweight and appropriate for constrained IoT environments. In 2018, several studies were published on remote user authentication for the IoT environment [36–41]. The author in [36] presented an authentication scheme for ad hoc WSN to improve the security weakness of the scheme in [42] using ECC cryptography. In Cyber-Physical Systems (CPS) and IoT, Lu et al. [37] presented a mutual authentication proposed scheme with user anonymity. Xu et al. [38] proved that Srinivas et al.’s [43] authentication schemes are vulnerable by many attacks and did not achieve user anonymity features. Moreover, Ryu et al. [39] reviewed Wu et al.’s [44] scheme and pointed out that Wu et al.’s scheme has two security weaknesses against outsider attackers. A new user authentication scheme was presented by Wazid et al. [40] for a hierarchical IoT network. These authors observed that their scheme involved lower computation and communication costs. Moreover, Chen et al. [41] presented an authentication scheme based on the fuzzy extractor. Nonetheless, the overhead in Chen’s scheme is costly.

More recently, in 2019, articles on this subject were published by Dammak et al. [45], Gupta et al. [46], Lyu et al. [47], Ma et al. [48], Renuka et al. [49], and Li et al. [50]. However, these schemes still have weaknesses, particularly in terms of the computation and communication overheads, which are highly compared to those of our proposed scheme. In summary, most remote user authentication protocols either fail to achieve IoT WSN environment security requirements or they do not provide security functionality features such as dynamic anonymity and untraceability and biometric and password change procedures. To overcome the aforementioned weaknesses, we proposed a lightweight remote user authentication protocol suited for the IoT WSN application, which achieves user anonymity and untraceability.
3. Basic Preliminaries

In this section, we briefly discuss the properties of the one-way hash function, perceptual hashing, and level 3 feature extraction.

(i) Level 3 feature extraction of fingerprint: a fingerprint is the pattern of ridges and valleys on the outer surface of the fingertip, and each individual has unique fingerprints. Fingerprint identification involves three levels: the first level includes details such as the pattern type and ridge-line flow; the second involves minutiae points for instant bifurcations, spurs, and terminations; and the third relates to the dimensional properties of a ridge, such as incipient ridges, creases, pores, and edge contours. The third level contains all the dimensional properties of a ridge for instant sweat pores, initial ridges, edge, and the crispiness. Our proposed protocol therefore adopts the third level, since it is unique, unalterable, and perpetual. More detail can be found in [51].

(ii) One-way hash function: one-way hash function (1HF) is a mathematical function that is broadly used in many applications, such as disclose data integrity during transmission, generating message authentication codes (MAC), and digital forensic investigations. Cryptographic 1HF is highly sensitive to even small perturbations to the input. The 1HF is impossible to invert, i.e., it is difficult to regain the original text from the hash value. It produces hash values of 128 bits and higher. Generally, 1HF is used to generate digital signatures, which are used to recognize and authenticate the sender [52].

(iii) Perceptual hashing: when using biometrics for user authentication schemes, the standard encryption or hashing algorithms cannot be used to encrypt the biometric template. The biometric data such as fingerprint and voice change with time and environment. Therefore, in designing a user authentication protocol using biometrics, the hashing or encryption algorithms cannot be utilized to encrypt the biometric template. To deal with this issue, researchers have proposed using perceptual hashing (p-hash) [53]. The advantage of using p-hash is capability tolerant to unimportant variation in the quality and format of the input. The hash value size that is generated by perceptual hashing differs from 64 to 128 bits [54]. In this paper, we adopted the perceptual hashing function proposed by Jie [55] in a previous study. The authors in [56] merge the image blocks which have low-frequency DCT coefficients and the color histogram as a perceptual feature, and this perceptual feature then compressed as interfeature with PCA and threshold the interfeature to generate a strong hash. Figure 2 shows the process of perceptual hashing.

![Figure 2: Perceptual hashing based on block-DCT and PCA based image.](image)

4. The Proposed Protocol

In this section, we propose an efficient and secure user authentication protocol for IoT WSN applications using the network model scenario presented in Figure 1. We also mention that the proposed protocol is designed to be generic enough for most IoT WSN applications that require user authentication.

A summary of the symbols used in this paper is given in Table 1. In this work, we utilize the current timestamps to ensure flexibility to replay attacks. In this work, we utilize the current timestamps to ensure flexibility to replay attacks. Consequently, the clocks of all protocol objects are assumed to be synchronized which is a typical assumption in the literature [7, 44, 57]. Our authentication protocol based on three factors, namely, password, user’s biometric, and smartphone focuses on the user in order to reduce the costs to the IoT nodes. Using a smart device such as a smartphone, the user can easily access the IoT nodes and the services they provide.

The proposed protocol contains three participants: a remote user ($U_i$) who aims to maximize the services in the environment, a set of IoT sensor nodes ($D_k$), and a trusted home authority/gateway (GW). Our work consists of four phases: registration, precomputation, authentication and key agreement, and password change phase. The registration phase was performed once, while the precomputation, authentication, and password change phase are executed whenever a remote user wishes to login or change his/her password. The proposed protocol enables the remote users to freely update his password and/or biometric information with the help of the smartphone without further involving GW.

4.1. Remote User Registration Phase. User side: at this stage, $U_i$ who aims to access IoT resources must initially register in the GW. To complete the registration, $U_i$ executes the following steps:

(Step 1) The $U_i$ selects his identity ($ID_i$) and password($PW_i$). Afterwards, $U_i$ inputs his/her fingerprint($FP_i$).
The \( U_i \) computes level 3 feature extraction of the fingerprint as follows: \( FX_3^{FP_i} = \text{FeatExt}(FP_i) \)

(Step 3) The \( U_i \) selects a random integer \( R_i \in \mathbb{Z}_n^+ \) and computes a mask for the user’s identity, password, and fingerprint as follows: identity mask: \( MID_i = H((ID_i \oplus R_i) \parallel X_{g\alpha}) \), password mask: \( MPW_i = H(PW_i \oplus R_i) \), and fingerprint mask: \( MFP_i = h(FX_3^{FP_i}) \)

(Step 4) The \( U_i \) sends \( MID_i, MPW_i, MFP_i, \) and \( FX_3^{FP_i} \) to the GW as a communication request to the GW node through a secure channel.

Gateway side: on receiving a request message from \( U_i \), the GW performs the following steps.

(Step 1) GW generates secret keys \( X_g \) and \( X_{g\alpha} \). Following this, GW computes the security parameters \( a_i = H(MID_i \oplus X_g) \), \( b_i = H(MPW_i \oplus X_{g\alpha}) \), and \( c_i = H(FX_3^{FP_i} \oplus X_{g\alpha}) \), prior to use

(Step 2) GW calculates Factor = \( \sum_{i=1}^L \text{ASCII}(FX_3^{FP_i}) \cdot e_i \), where \( e_i = a_i^i \oplus X_{g\alpha} \), and \( f_i = b_i^i \oplus X_{g\alpha} \)

(Step 3) The GW node submits \( M_1 = MID_i, e_i, f_i, X_{g\alpha}, \) and \( MFP_i \) to \( U_i \). On receiving \( M_1 \), \( U_i \) saves it in the memory of the device. Figure 3 summarizes the different processing steps followed during this phase.

4.2. IoT Sensor Node Registration. In this stage, each IoT sensor node is a registry. Any supplementary nodes can be added dynamically. This stage consists of the following steps.

IoT node side:

(Step 1) \( D_k \) generates a random number \( R_k \in \mathbb{Z}_n^+ \). \( D_k \) known the shared secret \( X_{g\alpha} \) of the GW and has a unique identity \( ID_{D_k}^* \). It sends \( R_k \) to the GW.

(Step 2) \( D_k \) computes the parameters \( MPW_j = h(X_{g\alpha} \parallel R_k \parallel ID_{D_k}^*) \) and \( MD_k = R_k \oplus X_{g\alpha} \oplus \text{Factor}_j \) for further calculation

(Step 3) \( D_k \) sends \( MPW_j, MD_k, ID_{D_k}^* \), and \( TS_1 \) to the GW through a secure channel.

GW node side: upon receiving the registration request from IoT sensor nodes \( D_k \), the GW calculates the following steps.

(Step 1) Checks the timestamp condition \( |TS_1 - T| < \Delta T \). If the condition is unsatisfied, then the registration phase is terminated; otherwise, the GW executes the next step

(Step 2) Computes \( R_i = MD_k \oplus X_{g\alpha} \oplus \text{Factor}_j \) on the basis of the previous message received from \( U_i \) as \( MPW_j = h(X_{g\alpha} \parallel R_i \parallel ID_{D_k}^*) \)

(Step 3) Verifies whether \( MPW_j = MPW_j^* \) or not. Therefore, if they are not unequal, the node is not

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( U_i )</td>
<td>Remote user</td>
</tr>
<tr>
<td>( ID_i )</td>
<td>Identity of the user</td>
</tr>
<tr>
<td>( PW_i )</td>
<td>The user password</td>
</tr>
<tr>
<td>( X_{g\alpha} )</td>
<td>Extracting user iris</td>
</tr>
<tr>
<td>( R_i )</td>
<td>Random numbers generated by the user</td>
</tr>
<tr>
<td>( FP_i )</td>
<td>The fingerprint of the user</td>
</tr>
<tr>
<td>( FX_3 )</td>
<td>Feature extraction level 3</td>
</tr>
<tr>
<td>( D_k )</td>
<td>IoT sensing nodes</td>
</tr>
<tr>
<td>( X_{g\alpha} )</td>
<td>The shared secret key of the IoT devices, generated by ( D_k )</td>
</tr>
<tr>
<td>( GW )</td>
<td>The trusted gateway node</td>
</tr>
<tr>
<td>( A )</td>
<td>A secret key generated by the gateway</td>
</tr>
<tr>
<td>( T )</td>
<td>A secret key generated for the user</td>
</tr>
<tr>
<td>( TS_1, TS_2, TS_3, TS_4, T )</td>
<td>Timestamps</td>
</tr>
<tr>
<td>( H(\cdot) )</td>
<td>One-way hash function</td>
</tr>
</tbody>
</table>

Table 1: Symbols used.
illegitimate, and the GW terminates the session. Otherwise, GW executes the next step

(Step 4) Computes the following parameters for further use, \( a_j = h(ID_j \parallel X_{g_d}) \), \( b_j = h(a_j \parallel X_{g_d}) \), and \( c_j = H(FX3_{FP}) \). Factor = \( \sum_{i=1}^{L} ASCII(FX3_{FP}) \). 

(Step 5) Then, GW sends \( a_j, c_j, \) and \( TS_2 \) to \( D_k \). Upon receiving the registration messages (\( a_i, c_i, \) and \( TS_2 \)) from the GW, \( D_k \) checks the timestamp condition \( |TS_2 - T| < \Delta T \) to verify for any external interference. If the condition is unsatisfied, then the session is terminated; otherwise, \( D_k \) saves the parameters \( a_i, c_i, \) and \( TS_2 \) into its device memory. Finally, the user registration phase is accomplished. Figure 3 shows the steps of the IoT sensor node registration phase. 

4.3. Precomputation and Login Phase. Once the registration is accomplished successfully, an authorized user \( U_i \) can access any desired sensor node within the IoT network through the authentication phase. To start with the authentication phase, \( U_i \) must login to the selected IoT service application, following the login steps that are implemented during this phase.

(Step 1) First, the user \( U_i \) uses the smartphone to open the applications and enters his/her password \( PW_i \) and level 3 feature extraction \( F_X_3 \) saved in the smartphone.

(Step 2) Then, the smartphone of \( U_i \) calculates a masked for the password and the feature extraction as follows: \( MPW_i = h(PW_i \bigoplus R_i) \) and \( MFP_i = h(FX3_{FP} \bigoplus X_{g_d}) \). Also, it computes \( b_{i}^* = h(a_i \parallel X_{g_d} \parallel MPW_i \parallel TS_2) \) and \( c_{i}^* = h(MFP_i \parallel X_{g_d} \parallel TS_2) \).

(Step 3) Next, the original values of \( b_i \) and \( c_i \) extract as follows: \( b_i = h(MPW_i \bigoplus X_{g_d}) \) and \( c_i = h(MFP_i \bigoplus X_{g_d}) \).

(Step 4) \( U_i \) computes the value of the following verification parameters \( b_{i}^{**} \) and \( c_{i}^{**} \) as follows: \( b_{i}^{**} = h(a_i \parallel MPW_i \bigoplus X_{g_d} \bigoplus Factor_i) \) and \( c_{i}^{**} = h(MFP_i \bigoplus X_{g_d} \bigoplus Factor_i) \).

(Step 5) Then, the accurateness of \( b_{i}^{**} \) and \( c_{i}^{**} \) is verified with \( b_{i}^{**} = b_{i}^{*} \) and \( c_{i}^{**} = c_{i}^{*} \).
The IoT sensor node $D_k$ must be successfully verified by the GW on the basis of the retrieved (Factor,) depending on MID. Therefore, GW performs the following:

1. **GW calculates $R_{*}^{k} = U_{i} \oplus h(MID_{i} \oplus X_{g})$ and $UD_{k} = h(X_{g} \| TS_{i} \| \text{Factor.} \| R_{*}^{k})$**
2. **GW compares the original $UD_{k}$ and the calculated $UD_{k}'$ to authenticate the $U_{i}$. If the verification condition is unsuccessful, then the GW terminates the communication; otherwise, the GW continues to the next step.
3. **Next, GW computes the security parameters: $GP_{ij} = \text{Factor.}^{*} \oplus R_{i} \oplus h(a_{j}^{*} X_{g_{0}})$ and $V_{i} = h(U_{i} \| TS_{i} \| TS_{k} \| X_{g_{0}})$**
4. **GW submits $M_{j} = \{GP_{ij}, V_{i}, TS_{j}\}$ to $D_{k}$**

Upon receiving the verification parameters $M_{j}$ from the GW, $D_{K}$ computes the following processes:

1. **$D_{K}$ verifies the timestamp $|TS_{j} - T| < \triangle T$. If the verification condition is unsatisfied, then the process is terminated; otherwise, it continues forward.**
2. **Then, $D_{K}$ checks the validity of $GP_{ij}$ with $\text{Factor.}^{*} \oplus R_{i} \oplus h(a_{j}^{*} X_{g_{0}})$. If the condition is unsatisfied, then the process is terminated; otherwise, it proceeds to the next step.**
3. **Next, $D_{K}$ calculates $cha = R_{i} \oplus \text{Factor.} \oplus M_{i}$, and $V_{*}^{k} = h(V_{i} \| TS_{i} \| TS_{k} \| X_{g_{0}})$**
4. **At last, $D_{K}$ sends $M_{*} = \\{V_{*}^{k}, TS_{j}, TS_{i}, TS_{k}, TS_{6}, cha\}$ to the $U_{i}$. When $U_{i}$ receives the verification parameters $M_{*}$ from $D_{K}$, $U_{i}$ executes the following steps:**
   1. **$U_{i}$ performs timestamp checks, i.e., $|TS_{6} - T| < \triangle T$? If not, then the process is terminated; otherwise, it continues to the next step**
   2. **$U_{i}$ retrieves $M_{*} = cha \oplus R_{i} \oplus \text{Factor.}^{*}$**
3. **Then, $U_{i}$ computes $V_{*}^{*} = h(U_{i} \| TS_{i} \| TS_{k} \| TS_{6} \| M_{*})$. Then, $U_{i}$ verifies if $(V_{*}^{*} = V_{*}^{k})$. If not, then $U_{i}$ is unsure of the authority of $D_{K}$ and the GW; otherwise, the $U_{i}$ computes the session key as $SK = h(M_{*}^{*} \oplus R_{i} \oplus \text{Factor.}^{*})$ and the authentication and key agreement phase successfully.

4.4. Authentication and Key Agreement Phase. To access the services of the IoT sensor nodes, the user will attempt to login to the proper node, after which the node will redirect the user login request to GW, which will carry out the necessary process to check the user’s authentication. When mutual validation is achieved between these three entities, a session key will be established between the user and the IoT sensor node. Figure 4 summarizes the login and the authentication phase. The following steps illustrate the processes of this phase.

1. **On receiving the login request message from $U_{i}$, $D_{k}$ performs the timestamp check on receive $TS_{i}$, i.e., $(|TS_{i} - T| < \triangle T)$. Also, check the security parameter $(\text{Factor.}^{*} = \text{Factor.} \oplus TS_{i})$, to authenticate the $U_{i}$. If the condition is unsatisfied, then the login is terminated; otherwise, the process proceeds to the next step.**
2. **$D_{k}$ uses the stored values of $e_{j}$ and $a_{j}$ to calculate $b_{j} = e_{j} \oplus a_{j}$**
3. **Next, $D_{k}$ calculates $A_{j} = h(X_{g_{0}} \| TS_{1} \| TS_{2}) \oplus b_{j}$**
4. **$D_{k}$ sends $M_{6} = \{U_{C_{i}}, UD_{k}, e_{j}, f_{j}, TS_{3}, TS_{4}, \text{Factor.}^{*}\}$ to the GW. GW can recognize the legitimacy of the $U_{i}$ and the node $D_{k}$ on the basis of the parameter Factor. and the transaction time. In this step, the node $D_{k}$ authenticates the GW.**
5. **GW verifies the received timestamp $(|TS_{i} - T| < \triangle T)$ and $(\text{Factor.}^{*} = \text{Factor.} \oplus TS_{i})$, authority of the $U_{i}$, and the device $D_{k}$ simultaneously. If the condition is satisfied, then the GW proceeds to the next step; otherwise, the process is terminated.**
6. **Then, GW calculates the security parameters: $a_{j}^{*} = h(Id_{D_{k}} \| X_{g})$, $b_{j}^{*} = e_{j} \oplus a_{j}^{*}$, and $b_{j} = A_{j} \oplus h(X_{g_{0}} \| TS_{1} \| TS_{2})$. Afterwards, the GW checks the quality of $b_{j}$ and $b_{j}^{*}$. If they are equal, then GW authenticates the node $D_{k}$ and the user $U_{i}$**
(Step 1) $U_i$, who must change his/her password opens the IoT application on a smart device and enters his/her old password $PW_i$ and feature extraction $MFP_i$; then, he/she calculates the masked for each of and feature extraction of user biometrics as $MPWi = h(MPW_i \oplus R_i)$, $MFP_i = h(MFP_i \oplus R_i)$, $b_i^* = h(MPW_i \oplus X_{ga})$, $c_i^* = h(MFP_i \oplus X_{ga})$, $b_i^{**} = h(a_i^{Factor} \oplus e_i \oplus MPWi \oplus X_{ga})$, $c_i^{**} = h(MFP_i \oplus a_i^{Factor} \oplus Factor_i)$. Checks: $b_i^{**} = b_i^*$ and $c_i^{**} = c_i^*$?

Computes: $UD_i = H(X_{ga} \mid TS_i \mid Factor_i \mid R_i)$, $UC_i = R_i \oplus a_i$, $Factor_i^* = Factor_i \oplus TS_i$, Sends: $M_2 = \langle MID_i, UD_i, UC_i, TS_i \rangle$

Checks: $|TS_i - T| \leq \Delta T$, $K_i^* = V_i^* \oplus H(V_i \mid T_{m_i})$, Computes $V_i^{**} = h(UD_i \mid TS_i \mid TS_i \mid TS_i \oplus K_i^*)$, Verify if $(V_i^{**} = V_i^*)$, if true; Computes: $SK = h(K_i^* \oplus R_i)$

(Step 2) Then, $U_i$ calculates $b_i^{**} = h(a_i^{Factor} \oplus e_i \oplus MPWi \oplus X_{ga})$ and $c_i^{**} = h(MFP_i \oplus b_i \oplus Factor_i \oplus Factor_i)$. (Step 3) Next, $U_i$ checks the equality of $b_i^{**}$ and $c_i^{**}$ with the original one $b_i^* = h(MPW_i \oplus X_{ga})$ and $c_i^* = h(MFP_i \oplus X_{ga})$. If any conditions do not hold, the $U_i$ unsuccessfully enters his correct data, and the system will be terminated; otherwise, $U_i$ is a valid user, and then $U_i$ is permitted to change his/her password.

(Step 4) $U_i$ enters the new password $PW_i^*$ and new fingerprint $FP_i^*$. Then, $U_i$ calculates mask the hash function for each of them as $MPWi^* = h(PW_i^* \oplus R_i)$ and $MFP_i^* = h(FP_i^* \oplus R_i)$, correspondingly.

(Step 5) Then, $U_i$ updates the parameter $b_i^*$ on the basis of the new password as $b_i^* = h(MPW_i^* \oplus X_{ga})$. Then, it calculates new $f_i$ as: $f_i^* = b_i^* \oplus X_{ga}$. (Step 6) At last, $U_i$ changes the old $f_i$ stored in the smart device memory with the new one $f_i^*$, and the phase terminates successfully.
5. Security Analysis

We evaluate the security strength of the proposed protocol using both formal and informal security analysis in this section. First, we prove that the proposed protocol provides mutual authentication between the remote user and the IoT sensor node using the BAN logic verification. First, we prove that the proposed protocol provides mutual authentication between the remote user and the IoT sensor node using the BAN logic verification. Then, we prove that the proposed protocol is resistant to other well-known attacks using informal security analysis. After that, we perform a formal security analysis using the popular widely accepted automated verification tool, AVISP.

5.1. Mutual Authentication Proof through BAN Logic. We use the widely recognized BAN logic [58] to prove that the mutual authentication is achieved between the registered legitimate remote user and an accessed IoT sensor node with the help of a trusted gateway node. Table 2 shows the symbols used of BAN logic and their respective abbreviations, where P and Q represent the principals, and X denotes a statement.

There are five rules used which govern the BAN logic are listed as follows:

Rule 1: message meaning rules: $P | \equiv P$, $P |\equiv \langle X \rangle_k / P$ $|\equiv Q | \sim X$ OR $P |\equiv P \equiv Q, P |\equiv \langle X \rangle_k / P |\equiv Q | \sim X$

Rule 2: the nonce-verification rule: $P |\equiv \#(X), P |\equiv Q | \sim X / P |\equiv Q | \equiv X$

Rule 3: the jurisdiction rule: $P |\equiv Q | \equiv X, P |\equiv Q | \equiv X / P |\equiv X$

Rule 4: the freshness rule: $P |\equiv \#(X), P |\equiv Q | \equiv X$

Using the above rules, we the following prove Theorem.

**Theorem 1.** The proposed protocol provides secure mutual authentication between $U_i$ and $D_k$ in the presence of the GW.

**Proof.** We define the following four goals:

- **Goal 1:** $U_i |\equiv N_k |\equiv (U_i, D_k)$
- **Goal 2:** $U_i |\equiv (U_i, D_k)$
- **Goal 3:** $D_k |\equiv (U_i, D_k)$
- **Goal 4:** $D_k |\equiv (U_i, D_k)$

The idealization form of the transmitted messages during the login and authentication phase under the proposed protocol is presented as follows:

(M1) $U_i \overset{\text{via} D_k}{\rightarrow} GW: \{ID_i, T_m, (U_i, D_k)\}_{X_k}

(M2) $U_i \overset{\text{via} D_k}{\rightarrow} GW: \{UD_k, UC_i, e_i f_i, T_m, T_m, D_k \overset{\text{via} D_k}{\rightarrow} GW, D_k \overset{\text{via} D_k}{\rightarrow} GW\}_{X_k}

(M3) $D_k \rightarrow GW: \{SID_i, TS, \text{Factor}, D_k \overset{\text{via} D_k}{\rightarrow} GW\}_{X_k}

(M4) $D_k \rightarrow GW:

(M5) $GW \rightarrow D: \{TS_3, D_k \overset{\text{via} D_k}{\rightarrow} GW\}_{X_k}

(M6) $GW \rightarrow D: \{TS_3, TS_5, T_m, SP_i, D_k \overset{\text{via} D_k}{\rightarrow} GW\}_{X_k}

(M7) $U_i \overset{\text{via} GW}{\rightarrow} D_k:

(M8) $GW \overset{\text{via} D_k}{\rightarrow} U_i:

(M9) $D_k \rightarrow U_i: \{TS_3, TS_5, T_m, SP_i, V_i, D_k \overset{\text{via} D_k}{\rightarrow} GW, (U_i, r_i \overset{\text{via} D_k}{\rightarrow} GW\}_{X_k}$

We consider the following initial assumptions according to the proposed protocol description:
(H1) \( GW \models (U_i \xrightarrow{X_g} GW) \)
(H2) \( GW \models \#(T_{s_1}) \)
(H3) \( GW \models U_i \models (U_i \xrightarrow{ID_i} GW) \)
(H4) \( GW \models (U_i \xrightarrow{\text{SID}_{i} \oplus h(ID_{i} \oplus R, r_{i})} GW) \)
(H5) \( GW \models \#(r_i) \)
(H6) \( GW \models U_i \models (U_i \xrightarrow{r_i} GW) \)
(H7) \( GW \models (U_i \xrightarrow{X_{u_0}} GW) \)
(H8) \( GW \models \#(T_{s_2}) \)
(H9) \( GW \models D_k \models (D_k \xrightarrow{ID} GW) \)
(H10) \( GW \models (D_k \xrightarrow{X_{u_0}} GW) \)
(H11) \( GW \models \#(r_i') \)
(H12) \( GW \models N_k \models (D_k \xrightarrow{r_i'} GW) \)
(H13) \( D_k \models (D_k \xrightarrow{X_{u_0}} GW) \)
(H14) \( D_k \models \#(T_{s_3}) \)
(H15) \( GW \models D | \models (D_k \xrightarrow{X} GW) \)
(H16) \( D_k \models \#(r_i') \)
(H17) \( D_k \models GW \models (D_k \xrightarrow{r_i'} GW) \)
(H18) \( D_k \models U_i \models (U_i \xrightarrow{K_i} GW) \)
(H19) \( U_i \models (U_i \xrightarrow{\text{SID}_{i} \oplus h(ID_{i} \oplus R, r_{i})} GW) \)
(H20) \( U_i \models \#(K_i) \)
(H21) \( U_i \models GW \models (U_i \xrightarrow{K} GW) \)
(H22) \( U_i \equiv \#(T_{m_1}) \)
(H23) \( U_i \equiv D_k \models (U_i \xrightarrow{K} GW) \)

By analyzing the messages M1-M9 and assumptions H1-H23 based on the BAN logic rules, the goals (goal 1-goal 4) are provided as follows:

From M1, we get S1: \( GW \models \langle ID_{s_0}, T_{m_1}, (U_i \xrightarrow{ID_i} D_k) \rangle \)

Based on S1, H1, and rule 1, we get S2: \( GW \models \equiv (ID_{s_0}, T_{m_1}, (U_i \xrightarrow{ID_i} D_k)) \)

Based on H2 and rule 1, we get S3: \( GW \models \equiv (ID_{s_0}, T_{m_1}, (U_i \xrightarrow{ID_i} D_k)) \)

Based on S2, S3, and rule 2, we have S4: \( GW \models (U_i \xrightarrow{ID_i} GW) \)

Based on S4 and rule 5, we get S5: \( GW \models (U_i \xrightarrow{ID_i} GW) \)

Based on S5, H3, and rule 3, we can get S6: \( GW \models (U_i \xrightarrow{ID_i} GW) \)

From M2, we get S7: \( GW \models \langle UD_{k}, UC, \text{Factor}_{i}, e_i, f_i, \text{SID}_{i}, (U_i \xrightarrow{ID} D_k) \rangle \)

Based on S7, H4, and rule 1, we get S8: \( GW \models (U_i \xrightarrow{ID} GW) \)

Based on H2, H5, and rule 4, we could get S9: \( GW \models \#(U_i \xrightarrow{ID} GW) \)

Based on S5, S6, S10, and rule 5, we could get S12: \( GW \models \equiv (U_i \xrightarrow{ID_i} GW) \)

Based on S13, H7, and rule 1, we get S14: \( GW \models \equiv (U_i \xrightarrow{ID_i} GW) \)

Based on S14, S15, and rule 2, we get S16: \( GW \models \equiv (\text{SID}_{i}, T_{m_2}, \text{Factor}_{i}, (N_k \xrightarrow{ID_i} GW)) \)

Based on S16 and rule 5, we have S17: \( GW \models \equiv (D_k \xrightarrow{ID_i} GW) \)

Based on S17, H9, and rule 3, we could get S18: \( GW \models \equiv (D_k \xrightarrow{ID_i} GW) \)

From Mssg4, we get S19: \( GW \models \langle UD_{k}, UC, e_i, f_i, T_{m_1}, T_{m_2}, \text{Factor}_{i}, (D_k \xrightarrow{ID_i} GW) \rangle \)

Based on S19, H10, and rule 1, we get S20: \( GW \models \equiv (D_k \xrightarrow{ID_i} GW) \)

Based on H8, H11, rule 4, we have
Based on S20, S21, and rule 2, we have S22: GW | ≡ \[D_k | \equiv (D_k \leftrightarrow GW)\]

Based on S23, H12, and rule 3, we get S24: GW | ≡ (D_k \leftrightarrow GW)

Based on S25, H13, and rule 1, we have S26: D_k | \equiv GW | \neg \langle TS_3, (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on S28 and rule 5, we could get S29: D_k | \equiv GW | \equiv (D_k \leftrightarrow GW)

Based on S29, H15, and rule 3, we have S30: D_k | \equiv (D_k \leftrightarrow GW)

Based on S30, S31, H13, and Rule 1, we get S32: N_k | \equiv GW | \neg

\langle TS_1, TM_2, TS_3, SP_{ij}, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on H14, H16, and rule 4, we obtain the following: S33: D_k | \equiv #

\langle TS_1, TS_2, TS_3, SP_{ij}, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on S32, S33, and rule 2, we have the following:

S34: D_k | \equiv GW | \equiv \langle TS_1, TS_2, TS_3, SP_{ij}, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on S17, S18, and S34, we get S35: D_k | \equiv GW | \equiv (D_k \leftrightarrow GW)

Based on S35, S17, and H12, and rule 3, we get S36: D_k | \equiv (D_k \leftrightarrow GW)

From Mssg7, we get S37: D_k | \equiv (# \langle TS_3, r^*_i, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on S37, H13, and rule 1, we get: S38: D_k | \equiv U_i | \neg \langle TS_3, r^*_i, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on H14, H16, and rule 4, we get S39: D_k | \equiv (# \langle TS_3, r^*_i, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

From S38, S39, and rule 2, we get S40: D_k | \equiv U_i | \equiv \langle TS_3, r^*_i, (D_k \leftrightarrow GW), (D_k \leftrightarrow GW) \rangle_{x_{stn}}

Based on S17, S18, S35, and rule 5, we get goal 4: D_k | \equiv U_i | \equiv (U_i \leftrightarrow D_k)

Based on S36, H18, goal 4, and rule 3, it will lead to the following:

Goal 3: D_k | \equiv (U_i \leftrightarrow D_k)

From Mssg8, we get S41: U_i | \equiv \langle TS_3, r^*_i, (U_i \leftrightarrow GW), (U_i \leftrightarrow GW) \rangle_{x_{stn}}

Based on S41, H19, and rule 1, we get S42: U_i | \equiv GW | \neg \langle TS_3, r^*_i, (U_i \leftrightarrow GW), (U_i \leftrightarrow GW) \rangle_{x_{stn}}

Based on H20, and rule 4, we get S43: U_i | \equiv (

\langle TS_3, r^*_i, (U_i \leftrightarrow GW), (U_i \leftrightarrow GW) \rangle_{x_{stn}}

Based on S42, S43, and rule 2, we get
Proposition 3. Resistance to impersonation attacks.

Proof. In our proposed protocol, the attacker cannot extract or impersonate the level 3 feature extraction of the fingerprint of $U_i$. Moreover, if a malicious attacker attempts to adjust the parameter $UD_k = h(X_{i,k} || TS_i || Factor_i || R_i)$ to a new one as $UD_k'$, then the attacker will fail in the GW side due to a mismatch with $UD_k$ calculated by the GW in the authentication phase with $UD_k$. Therefore, the proposed protocol resists impersonation attacks.

Proposition 4. Resistance to replay attacks.

Proof. Assuming that a malicious attack aims to retransmit a message gained by eavesdropping on an efficient communication channel between the $U_i$ and the $D_k$ through the login and authentication phase, the attacker will fail, because our proposed protocol uses timestamps $(TS_i, TS_{i0}, TS_{i1}, TS_{i2})$, and the delay time of the timestamp is brief. Our proposed scheme also uses Factor* which is stored in the basis of level 3 feature extraction. Therefore, the proposed protocol provides an efficient security against replay attacks.

Proposition 5. Resistance to stolen smart device attacks.

Proof. In the case where the user’s smart device is stolen or lost, the attacker aims to access the sensitive information stored in the device’s memory using a power examination attack. Our proposed protocol provides efficient security against this kind of attack. The attacker cannot determine the identity $ID_i$ and the password $PW_i$ of the $U_i$ since these are masked by a hash function on the basis of a random number $R_i$ that is generated only once. Moreover, the attacker cannot identify the feature extraction $FP_i$ given the hash function. Accordingly, our proposed protocol provides an efficient security against stolen smart device attacks.

Proposition 6. Resistance to password change attacks.

Proof. To change the user password, a malicious attack must use the personal fingerprint of a genuine $U_i$. Thus, the attacker cannot change the password. Assuming that a user’s smart device is stolen or lost or is used by an attacker through another method, the attacker still cannot change the password, since this process requires the old password. Therefore, our proposed protocol resists password changes attacks.


Proof. In our proposed scheme, this kind of attack is infeasible because the $U_i$ receives an authorization message from the node $D_k$ for security verification. Furthermore, we use timestamps $(TS_i, TS_{i0}, TS_{i1}, TS_{i2})$ to mitigate any crucial request.
Proof. If a malicious attack aims to build a parallel session of the scheme, then the attacker will fail even if he/she interrupts the communication message \((M_5 = \{\text{Factor}'_i, UDk_i, UC_i, TS_i, f_i\})\) due to our utilization of level 3 feature extraction and the timestamp used in the login and authentication phases.

**Proposition 9. Resistance to gateway node bypass attacks.**

Proof. In this kind of attack, the attacker aims to impersonate the GW with the aim of later connecting to any IoT node \(D_k\). Our proposed scheme can resist this type of attack because, as illustrated in Figure 4, the \(U_i\) initially sends the authenticated message \(\text{Factor}'_i, UDk_i, UC_i, TS_i, f_i\) to the desired node \(D_k\) to initiate the authenticated phase. Subsequently, the node \(D_k\) returns this message to the GW (Figure 3.7). Then, GW verifies \(D_k\) and \(U_i\). Accordingly, \(U_i\), as the first step, uses any IoT facility that is disconnected from the GW. Therefore, the proposed protocol provides an efficient security against gateway node bypass attacks.

**Proposition 10. Resistance to MITM attacks.**

Proof. As previously explained in Section 4, this type of attack aims to intercept communication between two legitimate parties and to modify, delete or delay messages. We suppose that a malicious attack intercepts the login message \((M_5 = \{\text{Factor}'_i, UDk_i, UC_i, TS_i, f_i\})\) transmitted from \(U_i\) to the node \(D_k\) and the authentication message \((M_6 = \{UC_i, UDk_i, f_i, TS_i, TS_a, \text{Factor}'_i\})\) that transmitted from the node \(D_k\) to GW. In this scenario, the attacker aims to modify the login message or authentication message to \((M_5', M_6')\). However, the attacker cannot predict the shared secret key needed to modify these messages. Moreover, each message communicated in the login and authentication phases has a timestamp with a short delay, thereby preventing an attacker from changing the messages. Therefore, our proposed scheme resists MITM attacks.

**Proposition 11. Resistance to Off-line Guessing Attacks.**

Proof. In our proposed scheme, a malicious attacker cannot gain an advantage by using off-line password-guessing attacks because the attacker cannot obtain the real passwords of a genuine \(U_i\) using the communication messages \(M_5 = \{\text{Factor}'_i, UDk_i, UC_i, TS_i, f_i\}\) and \(M_6 = \{UC_i, UDk_i, f_i, TS_i, TS_a, \text{Factor}'_i\}\) in the login and authentication phases, respectively. Even if the user’s smart device is stolen, the attacker cannot predict the password due to the nature of the hash function. Furthermore, the attacker cannot deduce the user fingerprint because the fingerprint is stored on the basis of a random number \(R_i\), level 3 feature extraction that is generated only once, and the use of a strong hash function. If the adversary guesses \(PW_i\) then, to legalize \(PW_i\) with \(b_i^* = b_i^{**}\) and \(c_i^* = c_i^{**}\), he/she needs to know \(U_i\)’s identity as well as \(U_i\)’s biometric \(B_i\). Moreover, to guess \(PW_i\), the adversary will need to guess \(ID_i\) and \(B_i\) along with the password. However, revealing of user biometric information or stealing it or forging it is not achievable; hence, the proposed protocol withstands offline-password-guessing attacks.

**Proposition 12. Provides key agreement.**

Proof. This feature indicates that the \(U_i\) and the IoT node \(D_k\) must agree on a secure session key to protect their successive communications. In our protocol, once \(D_k\) receives the authentication request \((M_7 = \{GP_i, V_i, TS_i\})\) from the GW, it computes the session key \(SK = H(R_i \oplus M_i \oplus \text{Factor}'_i)\) on the basis of mask nonce \(M_i\), \(R_i\) and \(\text{Factor}'_i\). Afterwards, \(D_k\) sends the message \((M_8 = \{V_i, TS_i, TS_a, TS_e, \text{cha}\})\) to the \(U_i\). Subsequently, \(U_i\) receives the authenticated message and then calculates the session key \(SK = h(M_i^* \oplus R_i)\), and both session keys are equal as shown in Figure 3.6. Therefore, our proposed protocol supports a secure session key.

**Proposition 13. Provides user anonymity.**

Proof. Anonymity means protecting the information of a \(U_i\) from being tracked by an attacker. The user information, identity \(ID_i\), password \(PW_i\), and fingerprint \(FP_i\) are masked by a hash function. The fingerprint \(FP_i\) is calculated on the basis of the hash function of level 3 feature extraction. Accordingly, if a particular attacker attempts to interrupt the message exchange between the entities, then the attacker will fail to trace the user information.

**Proposition 14. Provides forward secrecy.**

Proof. In our protocol, we created the session key \(SK = h(R_i \oplus M_i \oplus \text{Factor}'_i)\) on the basis of the nonce number \(M_i\), which is generated once for each \(U_i\) who desires to log in to the (IoT) nodes. We also created the random number \(R_i\) and the \(\text{Factor}'_i\), which is not saved as a plaintext. Therefore, a malicious attack cannot obtain the session key by any way.

**Proposition 15. Provides mutual authentication.**

Proof. An authentication mechanism requires each entity in the IoT environment, i.e., GW, the \(U_i\) and the IoT node \(D_k\) to validate each other. In our proposed protocol, after executing the necessary steps, \(U_i\) sends the authentication message \((M_5 = \{\text{Factor}'_i, UDk_i, UC_i, TS_i, f_i\})\) to \(D_k\) in the login phase (see Figure 3.7). Then, \(D_k\) send the authentication message \((M_6 = \{UC_i, UDk_i, TS_i, TS_a, f_i, \text{Factor}'_i\})\) to GW. Accordingly, the GW uses the authenticated message to validate the \(U_i\) and the node \(D_k\). Therefore, the proposed protocol achieves mutual authentication.

**Proposition 16. Provides Key Freshness**

Proof. In our work, we generate a session key \(SK = h(R_i \oplus M_i \oplus \text{Factor}'_i)\) that consists of fresh timestamps that are different in each session. Accordingly, our proposed protocol achieves key freshness. A security analysis of possible attacks
against this model is presented below, and it is shown that our proposed protocol can resist several well-known attacks.

5.3. Formal Security Analysis Using the AVISPA Tool. AVISPA is a powerful automated validation tool which provides a wide applications range for constructing cryptographic protocols analysis models, verification, and validation. To validate the protocol using the AVISPA tool, firstly, the protocol is coded by using HLPSL language. Then, translate the HLPSL code in intermediate format (IF) by the HLPSL2IF translator. Finally, the IF specification as input is given to the back ends. After the IF execution, the back-end displays the result of the simulation of the protocol by analyzing to output format (OF), with an explanation of whether the protocol is safe or unsafe against man-in-the-middle and replay attacks. Also, back ends confirm the security features of the protocol such as the flexibility against most of the known attacks, authentication, and the secrecy of keys. Note that AVISPA performs the Dolev-Yao threat model [59, 60]. More details of the AVISPA tool and HLPSL can be found in [61].

To implement and simulate the proposed protocol on AVISPA, we have concentrated on the major tool SPAN Version 1.6 based on a computer system which is consist of Windows 10 Enterprise operating system (64 bit) that is supported by Ubuntu 10.10 light on Virtual machine, Intel (R) Core (TM) i7-7500U CPU @ 2.70 GHz 2.90 GHz processor, and 8 GB RAM. In AVISPA, there is a role for each entity, and these roles are independent of each other. AVISPA has an implementation in the form of four back ends, namely, TA4SP (Tree Automata based on Automatic Approximations for the Analysis of Security Protocols), OFMC (On-the-fly Model-Checker), CL-AtSe (Constraint Logic-based Attack Searcher), and SATMAC (SAT-based Model checker) [62]. We have evaluated the proposed protocol against man-in-the-middle and replay attacks under the OFMC and CL-AtSe back ends using SPAN.

The user registration, login, and authentication phases for the proposed protocol are implemented in HLPSL utilizing three basic roles for a remote user, the IoT sensor node, and the gateway node. The compulsory roles for the environment, session, and goal are also defined. Figure 5 provides the simulation results that obviously indicate that the proposed protocol is protected against man-in-the-middle and replay attacks.

6. Comparative Study

The proposed protocol is compared with the recent user authentication protocols proposed in the IoT environment such as the protocols of Banerjee et al. [1], Yang et al. [8], Dhillon and Kalra [33], Dammak et al. [45], Li et al. [50], and Farkoon et al. [60].

6.1. Security Functionality Comparisons. Table 3 summarizes the comparison of the functionality features of the recent user authentication protocols [1, 8, 33, 45, 50, 60]. It can be observed that the proposed protocol offers improved security and functionality features, in comparison to the other recent protocols.

6.2. Computation Overhead Comparison. In this section, we compare our proposed protocol in terms of computation overhead with those of recent related protocols [1, 8, 33, 45, 50, 60]. The protocol comprises four phases: user and sensor node registration phase, login phase, key agreement and authentication phase, and password and biometrics change phase. In the IoT WSN environment, the performance of the user authentication protocol mainly is affected by the login and authentication phase [2]. These two phases are the major part of the user authentication protocol and is what chiefly characterize it from the different user authentication protocols in IoT WSNs. Consequently, we focused our discussion of computation overheads during the login and authentication phase. The computational costs are the time consumed by the user and service provider in the process [9]. For computation overheads analysis, we utilized the notations $T_b$ and $T_m$ to indicate the time complexity of the hash function and elliptic curve cryptography (ECC) algorithm, respectively. The computational costs of the XOR operation are usually neglected because it requires a minimal number of computations.

In the login and authentication phase of our protocol, the remote user requires only $87T_b$ to calculate the parameters of a login and authentication request message. The IoT sensor node expends only $37T_b$ bits to verify the login request and to calculate the parameters of the key agreement message. As for the gateway node, it requires the gateway node which requires only $77T_b$ bits to verify whether the verification equations hold. Our proposed protocol uses only the XOR and one-way hash function operations to design simple user authentication and key agreement protocol. However, Li et al.’s protocol [50] provides authentication and key agreement protocol that is designed using an asymmetric encryption ECC algorithm. The time complexity of the asymmetric ECC encryption operation is greater than that of a one-way hash function. According to the practical example of the computational costs in an environment with a CPU of 3.2 GHz and with 3.0 GB of RAM, the time complexity of one-way hash operations requires 0.02 ms when using SHA-1 and for the ECC encryption operation which requires 0.45 ms when using ECC-160 [63]. Therefore, the total computational overheads of our protocol are 0.36 ms. Table 4 summarizes the computational overheads of our proposed protocol and the existing protocols in [1, 8, 33, 45, 50, 60] with approximate time (in milliseconds). It is clear that the proposed protocol requires less overall computation costs. The energy consumption of the IoT sensor node in our work is 0.06 ms which is 50%, 81.25%, 25%, 87.75%, 62.5%, and 95.8% lower than the computation times in the protocols of [1, 8, 33, 45, 50, 60]. Consequently, the total energy consumed of the IoT sensor node by our protocol is 0.36 ms. Therefore, our proposed protocol is more efficient and suitable for constrained sensor devices in the IoT WSNs environment. Table 4 presents the energy consumption of the IoT sensor node of our proposed protocol with those of [1, 8, 33, 45, 50, 60] along with total
Table 3: Functionality comparison of our protocol with other recent related protocols.

<table>
<thead>
<tr>
<th>Properties</th>
<th>[1]</th>
<th>[8]</th>
<th>[33]</th>
<th>[45]</th>
<th>[50]</th>
<th>[60]</th>
<th>Our</th>
</tr>
</thead>
<tbody>
<tr>
<td>User anonymity</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mutual authentication</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Key agreement</td>
<td>✗</td>
<td>—</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to impersonation attack</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to MITM attack</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Replay attack</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to password guessing attack</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to GW bypassing attack</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to parallel session attack</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to smart device stolen attack</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to DOS attack</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to insider attack</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Password change phase</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Forward secrecy</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Key freshness</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BAN logic security analysis</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 4: Computation overhead comparison.

<table>
<thead>
<tr>
<th>Protocols</th>
<th>User</th>
<th>Sensor nodes</th>
<th>GW nodes</th>
<th>Total cost</th>
<th>Total estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>$4T_h$</td>
<td>$24T_h$</td>
<td>$5T_h$</td>
<td>$33T_h$</td>
<td>0.66 ms</td>
</tr>
<tr>
<td>[8]</td>
<td>$16T_h$</td>
<td>$16T_h$</td>
<td>$20T_h$</td>
<td>$52T_h$</td>
<td>1.04 ms</td>
</tr>
<tr>
<td>[33]</td>
<td>$9T_h$</td>
<td>$6T_h$</td>
<td>$7T_h$</td>
<td>$22T_h$</td>
<td>0.44 ms</td>
</tr>
<tr>
<td>[45]</td>
<td>$20T_h$</td>
<td>$8T_h$</td>
<td>$20T_h$</td>
<td>$48T_h$</td>
<td>0.96 ms</td>
</tr>
<tr>
<td>[50]</td>
<td>$13T_h + 3T_m$</td>
<td>$4T_h + 2T_m$</td>
<td>$8T_h + T_m$</td>
<td>$24T_h + 6T_m$</td>
<td>3.18 ms</td>
</tr>
<tr>
<td>[60]</td>
<td>$13T_h$</td>
<td>$4T_h$</td>
<td>$13T_h$</td>
<td>$30T_h$</td>
<td>0.6 ms</td>
</tr>
<tr>
<td>Ours</td>
<td>$8T_h$</td>
<td>$3T_h$</td>
<td>$7T_h$</td>
<td>$18T_h$</td>
<td>0.36 ms</td>
</tr>
</tbody>
</table>
improvements of computational costs of the present protocols. From Table 5, it can be observed that the computational costs for the resource-limited IoT sensing device in our protocol is less in comparison to that of the recent existing protocols. The proposed protocol achieves superior performance because it consumes less energy compared to recent related protocols and is highly efficient. As the sensors nodes deployed in IoT networks have low battery life, low storage, and limited processing capability, the energy consumption of these IoT sensor nodes must be optimized. The IoT sensor nodes energy depends on two factors: the number of cryptographic operations to be performed and the amount of data being transmitted. Our proposed protocol minimized the number of cryptographic computations, therefore, more data can be transmitted via IoT sensor nodes. For the evaluation of the proposed protocol, the workload is not taken into consideration. In the future, the proposed protocol will be executed for different workloads in Cloud computing and IoT environments.

6.3. Communication Overhead Comparison. For communication overhead computation, we assumed that the timestamp, hash digest (assuming SHA-1 hashing algorithm is applied), identity, a random nonce, and the secret key are 128 bits, while the ECC operations are 160 bits. There are four exchanges messages between $U_i$, $N_i$, and $GW$ in proposed protocol that are $M_5 = \{MID_i, UD_i, UC_i, TS_i\}$, $M_6 = \{UC_i, UD_i, TS_i\}$, $M_7 = \{GP_{ij}, V_i, TS_i\}$, and $M_8 = \{V_i^*, TS_{ij}, TS_{ji}\}$, where $MID_i$, $UD_i$, $UC_i$, $GP_{ij}$, and $V_i$ are the hash function output. $TS_{ij}$, $TS_{ji}$, and $TS_{ji}$ are timestamps. According to our above assumption, each parameter is 128 bits. Therefore, the communication overhead of the proposed protocol is $128 \times 13 = 1664$ bits. Table 6 summarizes the communication overheads and the number of exchanged messages for all protocols in addition to the proposed protocol. We observe that the proposed protocol obtains less communication overhead as compared to the protocols in [1, 8, 33, 45, 50, 60] and incurs greater overhead than the protocol in [60]. Although the protocol [60] bears the less overhead but did not achieve the desired functionality and security features such as resistance to impersonation attacks, password guessing attacks, DOS attacks, preserved user anonymity, and forward secrecy in contrast to our protocol which achieved all functionality and security features (see Table 3).

7. Conclusion

In this paper, we presented a secure and lightweight three-factor remote user authentication protocol designed for future IoT WSN application. The proposed protocol grants the legitimate remote user that mutually authenticates with the IoT sensor node through a trusted gateway node. The symmetric session key $SK$ is established by the end of successful mutual authentication between the user and the IoT sensor node for future secure communications. The security of the proposed protocol is formally using the popular widely accepted BAN logic. Furthermore, informal security verification demonstrates that the proposed protocol resists the most well-known attacks. The formal security using the AVISPA simulation is evaluated, and the results showed that our protocol is safe. Finally, the performance analysis comparison in terms of computation and communication overheads demonstrated that our protocol showed high efficiency and performance compared to those of recent related protocols and is more suitable for practical IoT WSN environments.

Data Availability

All data are available within the manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References


