Arab J Sci Eng (2017) 42:765–786
DOI 10.1007/s13369-016-2341-x

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Cryptanalysis and Extended Three-Factor Remote User Authentication Scheme in Multi-Server Environment

Preeti Chandrakar[1] · **Hari Om**[1]

Received: 6 April 2016 / Accepted: 13 October 2016 / Published online: 1 November 2016

Abstract Recently, Wen et al. have developed three-factor authentication protocol for multi-server environment, claiming it to be resistant to several kinds of attacks. In this paper, we review Wen et al.'s protocol and find that it does not fortify against many security vulnerabilities: (1) inaccurate password change phase, (2) failure to achieve forward secrecy, (3) improper authentication, (4) known session-specific temporary information vulnerability and (5) lack of smart card revocation and biometric update phase. To get rid of these security weaknesses, we present a safe and reliable three-factor authentication scheme usable in multi-server environment. The Burrows–Abadi–Needham logic shows that our scheme is accurate, and the formal and informal security verifications show that it can defend against various spiteful threats. Further, we simulate our scheme using the broadly known Automated Validation of Internet Security Protocols and Applications tool, which ensures that it is safe from the active and passive attacks and also prevent the replay and man-in-the-middle attacks. The performance evaluation shows that the presented protocol gives strong security as well as better complexity in the terms of communication cost, computation cost and estimated time.

Keywords Authentication · *AVISPA* · *BAN* Logic · Cryptanalysis · Three-factor

✉ Preeti Chandrakar
preet29.chandrakaar@gmail.com

Hari Om
hariom4india@gmail.com

1 Department of CSE, IIT(ISM), Dhanbad, Jharkhand 826004, India

1 Introduction

Authentication is a procedure which recognized legitimacy of the system/user. It is the most important security mechanism in the public domain to access several web-based services such as online banking, e-commerce, m-commerce and e-health. The authentication process may be categorized in two environments, i.e., single-server environment and multi-server environment. In single-server environment, for obtaining various types of applications from various servers, the user needs to register on a particular server [1–5]. In order to register on different servers, the user memorizes the several confidential information like identity and password. But, it is an arduous task for the user to memorize various identities and passwords. Therefore, the user uses same identity and password on different servers for his/her amenities. However, this is not a good habit of the user to use same confidential information on different servers because if an attacker got user's confidential information, then he/she can access all servers wherever the user has registered. To avoid these vulnerabilities, multi-server authentications have come as a dynamic platform, where the users can contact with any server using a single registration [6–30]. In multi-server platform, one needs to register with a registration center only, for accessing services from multiple servers rather that registering with each and every server. This is one of the most important benefits of multi-server environment.

In 2000, Ford and Kaliski [16] developed password-based authentication scheme in multi-server platform that circulates password among a number of servers. This scheme computes a secret key using a password. The attacker was unable to compute secret key until/unless all the servers are cooperating. This scheme is more computationally intensive because of public keys. Moreover, the user needs a trustworthy channel for communicating with the server. To overcome these limitations, in 2001 Jablon [17] developed password-based authentication protocol in multi-server without requiring public key and trustworthy channel. To review their work, the numerous multi-server environment-based authenticated schemes have been drawn by many researchers. In 2009, Liao et al. [18] presented password and smart card-based authentication protocol with the help of dynamic identity and proclaimed that their scheme equips all the security aspects. However, Hsiang et al. [19] search out that Liao et al.'s protocol is not ready to hold up user and server impersonation attack, registration center impersonation attack, insider attack and not reparable. Additionally, the scheme has no mutual authentication feature. To solve these shortcomings, they designed an enhancement scheme over Liao et al.'s scheme. But, Sood et al. [20] got flaws in Hsiang et al.'s protocol like replay attack, spoofing attack and smart card stolen attack, and also the password update or change phase is incorrect. For resolving all the aforementioned security vulnerabilities, Sood et al. projected an extended authentication protocol that relies on changeable identity in multi-server platform and declared that the protocol is capable of holding up several sorts of security barriers. Unfortunately, Li et al. [21] affirm that Sood et al.'s scheme is not defending to stolen verifier attack, stolen smart card attack and spoofing attack. Further, its authentication phase is incorrect.

Frequently, we have seen that the numbers of multi-server authentication protocols are suffered from off-line password guessing attacks. Normally, the user often used simple password which is easy to crack with the help of simple dictionary attacks because they have low entropy. To surmount the password guessing attack, biometric authentications have been proposed which are more reliable and secure methods. It also points out that biometric template provides higher security than traditional password. The biometric key is more secure and cannot be distributed anywhere. That is the reason, the biometric key is not breakable [7,8,10,22–33]. In 2014 , Mishra et al. [22] designed three-factor-based authentication protocol usable in multi-server platform. They divulged that their scheme is safe from all sort of wicked attacks. But, Lu et al. [23,24] have shown that their scheme is unprotected from user and server spoofing attack and also not have forward secrecy property. To avoid these security issues, Lu et al. projected two authentication schemes for multi-server environment [23,24] and proclaimed that it is secure, more efficient from other related existing schemes. Later on, Chaudhary et al. [25] extracted the security shortcomings in Lu et al.'s scheme [23] and have shown that the scheme is doubtable from the user impersonation attack. After that, Chaudhary [26] has done cryptanalysis of the scheme which is presented by Lu et al. [23,24] and demonstrated that Lu et al.'s protocol [24] is insecure from user impersonation attack and not facilitates user anonymity; however, Lu et al.'s scheme [23] is also apprehensible to user impersonation attack. To sort out these security problems, they projected an extended authentication protocol.

In 2011, Das et al. [27] developed multi-server-based authentication protocol along with biometric. They divulged that their enhanced scheme provides brawny authentication with the help of three factors. However, An [28] identified the security weaknesses in Das et al.'s protocol and revealed that their scheme is defenseless to the user or server masquerading threat, password guessing threat and insider attack, and also, it does not provide mutual authentication. To get rid of these security issues, An design a new three-factor authentication scheme. But, unfortunately Khan et al. [29] revealed that An's protocol does not hold up password guessing threat and impersonation threat. Furthermore, their scheme does not procure mutual authentication and user anonymity property. To eliminate these security vulnerabilities, Khan et al. designed an upgraded biometric-based authentication scheme and proclaimed that their scheme can withstand the entire security problems and also provides extra security features. But, in 2015, Wen et al. [30] have reviewed Khan et al.'s protocol and pinpoint that their scheme is not ready to protect the password guessing attack and user impersonation attack and also does not provide user anonymity. Since then, Wen et al. proposed an improved biometric-based authentication scheme to remove these security problems. In this article, we have found out that Wen et al.'s [30] scheme unfortified against the various security pitfalls. To resolve these security pitfalls, we present three-factor remote user authentication scheme in multi-server environment.

1.1 Threat Model

1. An attacker $\mathcal{A}$ can pilfer the smart card of a user and disentangle the confidential data from it using the power consumption analysis [34,35]. Then the attacker tries to get the user's password by some means using these disentangled data.
2. The $\mathcal{A}$ can obstruct the communication message between entrant entities (user, server, registration center) over the untrustworthy channel. After that, attacker easily replays and modifies the obstruct message.
3. The registered user can act as an adversary or vice versa, and privileged insider of the registration center can also act as an adversary or vice versa.
4. The intruder can succeed to guess password and identity individually, but guessing two confidential data at the same time is computationally infeasible.
5. The $\mathcal{A}$ cannot trap and update any messages via the trustworthy channel.

6. If the length of ID_i or PW_i is n characters, then the guessing probability of n characters is approximately $\frac{1}{2^{6n}}$ [5,7,10].

1.2 Motivation and Contribution

In recent times, the number of users depends on the various remote servers, for acquiring different kinds of applications from the server. Therefore, numerous multi-server-based remote authentication schemes have been suggested, but many of them do not secure against various security threats [6–30]. Therefore, we are motivated to propose a three-factor remote authenticated protocol. We provide the following contributions.

1. First, we analyze Wen et al.'s [30] protocol and pinpoint some security weaknesses such as inaccurate password change phase, failure to achieve forward secrecy, improper authentication, known session-specific temporary information attack, absence of smart card revocation and biometric update phase.
2. To solve these security barriers, we have designed three-factor authenticated scheme in multi-server platform.
3. We have proved that the presented scheme is precise through the *BAN* logic.
4. The formal and informal security verification certifies the presented protocol is able to defend from the various types of security barrier.
5. We perform the simulation using the predominantly known *AVISPA* tool.
6. The presented scheme is more suitable in the context of communication and computation overhead and estimated time (in Seconds) as compared to Wen et al. [30] and other protocols [9,11–14,23,24].
7. The presented scheme carries extra security aspects as compared to Wen et al. [30] and other relevant protocols [9,11–14,23,24].

1.3 The Formation of the Article

The formation of this article is summarized as follows. We have concisely elaborated the hash function concept in Sect. 2. In Sect. 3, we have briefly reviewed the Wen et al.'s protocol. In Sect. 4, we elaborate the security pitfalls of Wen et al.'s scheme. In Sects. 5 and 6, the proposed scheme is demonstrated and the validity of the proposed scheme using the *BAN* logic is proved. The informal security analysis and simulation by using *AVISPA* tool are presented in Sects. 7 and 8. The formal security analysis based on random oracle is discussed in Sect. 9. In Sect. 10, we have delineated performance comparison. Lastly, we have drawn conclusion in Sect. 11.

Table 1 Notations used in this paper

Symbol	Description
U_i	The user
ID_i	The identity of U_i
PW_i	The password of U_i
RC	Registration center
ID_R	Identity of RC
S_j	The server
SID_j	Identity of S_j
E_K/D_K	Symmetric key encryption and decryption algorithm
x	The secret key of the RC
$h(.)$	Non-invertible hash function
$H(.)$	Bio-hash function
$\exists$	There exists
$\mathcal{A}$	An attacker

2 Preliminary

The notion of non-invertible hash function is briefly described in this section.

Hash function

The non-invertible hash function is a secure function which means that it does not exist inverse. The non-invertible hash function takes input as arbitrary length and produces output as fixed length. The hash function is defined as h: $\{0, 1\}^* \rightarrow \{0, 1\}^l$, where $\{0, 1\}^*$ is the input of arbitrary length in the context of binary either o or 1, and $\{0, 1\}^l$ is the output of fixed length. There are some following properties which demonstrate the hash function elucidated in detail below.

Preimage Resistant Let us consider $x \in \{0, 1\}^*$ is given. Then, we can easily calculate y, i.e., y = h(x).

Second Preimage Resistant This is very difficult to calculate that $x' \in \{0, 1\}^*$, i.e., $h(x) = h(x')$, for a given input $x \in \{0, 1\}^*$ and $x \neq x'$.

Collision Resistant This is very hard to determine the pair $(x, x') \in \{0, 1\}^* \times \{0, 1\}^*$ such that $h(x) = h(x')$, where $x \neq x'$.

3 Brief Overview of Wen et al. Scheme

In this segment, we have scrutinized the Wen et al.'s protocol [30], which consist of the following four phases such as (1) registration (2) login (3) authentication and (4) password change. Their scheme has three entities like RC, S_j and U_i. In Table 1, we have delineated the notations utilized in the whole article.

3.1 Registration Phase

The U_i picks up a random number K and transmits the registration request $\{ID_i, PW_i \oplus K, B_i \oplus K\}$ to the RC through a trustworthy channel. The RC computes $f_i = \text{h}((B_i \oplus K) \| (PW_i \oplus K))$, $r_i = \text{h}(PW_i \oplus K \oplus B_i \oplus K) \oplus f_i = \text{h}(PW_i \oplus B_i) \oplus f_i$ and $e_i = \text{h}(ID_i \parallel x) \oplus r_i$ after acquiring a registration request from the U_i. RC saves the parameters $(ID_i, h(.), e_i, f_i)$ into the smart card's memory and dispatches it to U_i through a reliable channel. Upon obtaining the smart card from the RC, the user keeps the random number K in it. Ultimately, the smart card contains $(ID_i, h(.), e_i, f_i, K)$.

3.2 Login Phase

When user U_i wishes to get the services of the S_j, then the U_i inserts the smart card into the terminal and keys $\{ID_i, PW_i, B_i\}$. The smart card calculates $f_i^* = \text{h}((B_i \oplus K) \| (PW_i \oplus K))$ and compares $f_i^* = f_i$. If it holds, the smart card reader evaluates the login message (ID_i, ID_R, M_2, M_3) and transmits it to the S_j, where $r_i = \text{h}(PW_i \oplus B_i) \oplus f_i$, $M_1 = r_i \oplus e_i$, $M_2 = E_{M_1}(R_1, T_i)$, $M_3 = \text{h}(M_1 \parallel R_1 \parallel SID_i \parallel T_i)$, where T_i is time stamp and R_1 is a random number created by U_i; otherwise, the session is terminated.

3.3 Authentication Phase

After obtaining the login message (ID_R, ID_i, M_2, M_3), S_j executes the following steps to authenticate the user U_i.

Step 1 S_j computes $M_4 = \text{h}(K_{RS} \parallel ID_i \parallel SID_j \parallel M_2 \parallel M_3 \parallel T_s)$, $M_5 = E_{KRS}(ID_j, M_2, M_3, M_4, R_2)$ and the S_j sends the message (M_5, SID_j, T_s) to RC.

Step 2 Now, acquiring the message (M_5, SID_j, T_s) from S_j, the RC inspects the legitimacy of T_s. If T_s is not valid, then the RC rejects the request; otherwise, the RC calculates $K_{RS} = \text{h}(SID_j \parallel x)$ and $D_{K_{RS}}(M_5) = (ID_i, M_2, M_3, M_4, R_2)$. Then, the RC compares retrieving value M_4 with a computed value $\text{h}(K_{RS} \parallel ID_i \parallel SID_j \parallel M_2 \parallel M_3 \parallel T_s)$. If this condition holds, then the RC computes $K_{RU} = \text{h}(ID_j \parallel x)$ and $D_{RU}(M_2) = (R_1, T_i)$. Subsequently, the RC checks the legitimacy of T_i and verifies $M_3 = \text{h}(K_{RU} \parallel R_1 \parallel SID_j \parallel T_i)$. If this condition holds, then the RC produces a random number R_3 and computes $M_6 = E_{K_{RS}}(R_1, R_3, T_r)$, $M_7 = E_{K_{RU}}(R_1, R_2, R_3, T_r)$, $M_8 = h(K_{RS} \parallel ID_i \parallel SID_j \parallel M_6 \parallel M_7)$, where T_r is current time stamp. Then, the RC sends message (M_6, M_7, M_8) to the S_j.

Step 3 The S_j verifies that $M_8 = \text{h}(K_{RS} \parallel ID_i \parallel SID_j \parallel M_6 \parallel M_7)$. If this verification holds, then the S_j believes the legitimacy of the RC and computes $D_{K_{RS}}(M_6) = (R_1, R_3, T_r)$. Then, the S_j inspects the authenticity of T_r, and if T_r is not valid, then the S_j terminates the session; otherwise, the S_j evaluates $M_9 = E_{R_3}(R_1, R_2, T_s^{'}, T_r)$ and $SK = \text{h}(R_1 \parallel R_2 \parallel R_3)$ and dispatches (M_7, M_9) to the U_i, where $T_s^{'}$ is the time stamp and SK is the session key.

Step 4 After achieving the message (M_7, M_9) from the S_j, the U_i computes $D_{M_1}(M_7) = (R_1, R_2, R_3, T_r)$ and $D_{R_3}(M_9) = (R_1, R_2, T_s', T_r)$. Then, the U_i checks the legitimacy of the (T_s', T_r). If these values are valid then only user verifies whether T_r, R_1, and R_2 decrypted from M_7 equals to the values decrypted from M_9. If this verification fails, then the U_i stops the session; otherwise, the U_i sets the $SK = \text{h}(R_1 \parallel R_2 \parallel R_3)$ as a session key and shares with the S_j.

3.4 Password Change Phase

Step 1 The U_i puts the smart card inside terminal and inputs ID_i, PW_i and imprints B_i.

Step 2 The smart card enumerates $f^* = \text{h}((B_i \oplus) K) \parallel (PW_i^{old} \oplus K))$ and checks $f_i = f^*$. If this comparison fails, then the system spurns the U_i's request or else the smart card asks to enter new password PW_i^{new} and computes $r_i^* = r_i \oplus h(PW_i^{old} \oplus B_i) \oplus h(PW_i^{new} \oplus B_i)$, $e_i^* = \text{h}(ID_i \parallel x) \oplus r_i^*$. Then, the smart card renovates the old value of (r_i, e_i) with (r_i^*, e_i^*), respectively.

4 Cryptanalysis of Wen et al. Scheme

In this part, we discuss the security vulnerabilities of Wen et al.'s protocol that include inaccurate password change phase, failure to achieve forward secrecy, improper authentication, known session-specific temporary information attack, absence of smart card revocation and biometric update phase.

4.1 Inaccurate Password Change Phase

A challenging task in developing an authentication scheme is that it should provide precise and user-friendly password update facility. Wen et al.'s protocol does not support accurate and user-friendly password update as discussed below:

Step 1 In the password change phase, the smart card reader asks new password from the user after checking the legitimacy of the U_i. The U_i inputs new password PW_i^{new}, and smart card reader evaluates $r_i^* = r_i \oplus h(PW_i^{old} \oplus B_i) \oplus h(PW_i^{new} \oplus B_i)$ and $e_i^* = \text{h}(ID_i \parallel x) \oplus r_i^*$. Subsequently, the smart card reader replaces the value of (r_i, e_i) with (r_i^*, e_i^*) without updating f_i. The password change phase is successfully executed and updates the old password to new password.

Step 2 In the next login session, smart card computes $f_i^{**} = h((B_i \oplus K) \parallel (PW_i^{new} \oplus K))$ and checks if $f_i^{**} = f_i$. This verification does not hold because the value of f_i has not been updated after changing the password. Thus, the login request of the legitimate user is rejected. This login request (which is from a legitimate user) will ever be discarded unless the user re-registers with the registration center.

In Wen et al.'s protocol, the password and biometric verification will always fail in the login as well as password change phase because in password updation, the smart card reader just replaces (r_i, e_i) with (r_i^*, e_i^*) while keeping f_i unchanged. Thus, the password change phase is not precise, and as a consequence, the verification procedure in the login phase will always fail. Therefore, a legal user cannot get the services from a remote server.

4.2 Failure to Provide Perfect Forward Secrecy

One of the most essential properties of authentication scheme is forward secrecy. It ensures that even if the secret key of the participant's entity is leaked to the attacker $\mathcal{A}$, the confidentiality of the session key is not revealed from this exposure. We provide two instances to show that Wen et al.'s scheme [30] does not have this property.

- **Case 1**

Suppose that the secret key K_{RU}=h($ID_i \parallel x$) $= M_1$ is disclosed to the attacker by some means. He/she can get SK by executing the steps as given below:

Step 1 An adversary obstructs the message $\{M_7, M_9\}$, where $M_7 = E_{K_{RU}}(R_1, R_2, R_3, T_r)$ and $M_9 = E_{R_3}(R_1, R_2, T_s', T_r)$.

Step 2 The adversary decrypts the message M_7 by using the disclosed secret key K_{RU}, i.e., $M_7 = D_{K_{RU}}$ (R_1, R_2, R_3, T_r).

Step 3 After decrypting the message M_7, the attacker knows the value (R_1, R_2, R_3).

Step 4 The adversary can compute $SK = h(R_1 \parallel R_2 \parallel R_3)$ using the random number R_1, R_2 and R_3. Thus, SK can be obtained.

- **Case 2**

Suppose that an adversary gets the value of $K_{RS} = h(SID_j \parallel x)$. He/she can get the session key by executing the steps as given below:

Step 1 The adversary $\mathcal{A}$ intercepts the message $\{M_5, SID_j, T_s\}$ and $\{M_6, M_7, M_8\}$ in the same session, where $M_5 = E_{K_{RS}}(ID_i, M_2, M_3, M_4, R_2)$, $M_6 = E_{K_{RS}}(R_1, R_3, 4T_r)$, $M_7 = E_{K_{RU}}(R_1, R_2, R_3, T_r)$ and $M_8 = h(K_{RS} \parallel ID_i \parallel SID_j \parallel M_6 \parallel M_7)$.

Step 2 $\mathcal{A}$ decrypts the message M_5 and M_6 using the K_{RS}, i.e., $D_{K_{RS}}(M_5)$= $(ID_i, M_2, M_3, M_4, R_2)$ and $D_{K_{RS}}(M_6) = (R_1, R_3, T_r)$.

Step 3 After decrypting the message M_5 and M_6, an attacker gets all the random number R_1, R_2 and R_3.

Step 4 An attacker $\mathcal{A}$ can computes $SK = h(R_1 \parallel R_2 \parallel R_3)$ using these ephemeral secret values R_1, R_2 and R_3.

From the above discussion, we can say that Wen et al.'s protocol does not facilitate perfect forward secrecy.

4.3 Improper Authentication

In Wen et al.'s protocol [30], the authenticity of the request message $\{ID_i, ID_R, M_2, M_3\}$ is not certified by the server S_j. Therefore, the attacker gets an opportunity to change the login message and impersonates as U_i. After receiving the message from the attacker, the server directly sends this message to the RC without verifying it, which enhances the extra computation and communication cost, thus increasing the network congestion. The improper authentication is possible in Wen et al.'s protocol as discussed below.

Step 1 During the login phase, the adversary traps the login request message (ID_i, ID_R, M_2, M_3). The adversary produces a random number R_1' and also creates a secret key x'. Then, the adversary computes $M_1' = h(ID_i \parallel x')$, $M_2' = E_{M_1'}(R_1', T_i)$ and M_3'=h($M_1' \parallel R_1' \parallel SID_j \parallel T_i$). The adversary sends the falsified message (ID_i, ID_R, M_2', M_3') to the S_j.

Step 2 Upon acquiring the message from the adversary, S_j computes $M_4' = h(K_{RS} \parallel ID_i \parallel SID_j \parallel M_2' \parallel M_3' \parallel T_s)$, M_5'=(E_{KRS}(ID_i, M_2', M_3', M_4', R_2) and sends the message (M_5', SID_j, T_s) to RC.

Step 3 Upon obtaining the message $\{M_5', SID_j, T_s\}$ from the S_j, the RC inspects the legality of T_s. If T_s is not accurate, then RC refuses the request message; otherwise, RC computes $K_{RS} = h(SID_j \parallel)$ and $D_{K_{RS}}(M_5')$= $\{ID_j, M_2', M_3', M_4', R_2\}$. After that, RC computes $h(K_{RS} \parallel ID_i \parallel SID_j \parallel M_2 \parallel M_3 \parallel T_s)$ and compares this value to the M_4'. If both values are same, then RC computes $K_{RU} = h(ID_i \parallel x)$ and $D_{K_{RU}}(M_2')$ =(R_1', T_i). But, RC cannot decrypt M_2' because the attacker encrypts M_2' with the key $M_1' = h(ID_i \parallel x')$ and R decrypts M_2' with the key $M_1 = h(ID_i \parallel x)$. The encryption and decryption procedures are performed using different keys; the

RC cannot get the value of $\{R_1', T_i\}$, and hence, the RC terminates the session.

The aforementioned discussions show that the attacker impersonates as a legal user and sends forged messages to the S_j. The S_j transmits this forged message to the RC, without verifying it. This confuses the RC that S_j is a forged server, but S_j is actually a legal server.

4.4 Known Session-Specific Temporary Information Attack

In session key generation function, some ephemeral secret information is used. If this transient information is disclosed to the attacker by some method, then secrecy of SK will be leaked out. In Wen et al.'s protocol, the SK = $h(R_1 \parallel R_2 \parallel R_3)$, where R_1, R_2 and R_3 are random numbers generated by user, server and registration center, respectively. We have observed that the SK only depends on ephemeral secret information R_1, R_2 and R_3. If an attacker obtains these secret information R_1, R_2 and R_3 by some means, then he/she easily calculates the session key. Therefore, Wen et al.'s protocol cannot stop the known session key-specific temporary information attack.

4.5 Lack of Smart Card Revocation and Biometric Update Phase

The smart card revocation is an essential need in remote user authentication protocol. Unfortunately, when smart card is lost, then there should be some rule for avoiding the illegal use of lost/stolen smart card. But, in Wen et al.'s scheme, there is no such type of rule for revoking the lost/stolen smart card which provides offer to the attacker to behave as a genuine user. If somehow an attacker gets the lost/stolen smart card, then he/she accesses all the confidential parameters from the smart card using power consumption analysis. After that, an attacker acts as a legal user and tries to access the services. Therefore, the smart card revocation phase is very mandatory in the field of remote user authentication. But, Wen et al.'s scheme does not equip smart card revocation phase. Additionally, Wen et al.'s scheme does not provide the facility of updating the biometric. So, for accessing highly secure applications, an authentication scheme must be granted to change or update old password and own biometric of a legitimate user.

5 The Proposed Scheme

In this segment, we develop a secure remote user authentication protocol, which depends on three factors such as password, smart card and biometric. It consists of the following five phases: registration, login, authentication, password change and smart card revocation. There are three entities such as the user U_i, the server S_j and the registration center RC. The summary of login and authentication procedure is described in Table 2.

5.1 Server Registration Phase

In this subsection, there are some following steps performed.

Step 1 Initially, the server S_j picks up identity SID_j freely and transmits it to the registration center RC over a trustworthy channel.

Step 2 Upon obtaining the SID_j from the server S_j, the RC produces a random nonce N_j and calculates $K_{j1} = h(SID_j \parallel x)$, $K_{j2} = h(K_{j1} \parallel N_j \parallel y_s)$. The RC stores $\{SID_j, N_j\}$ in the database. After that, the RC sends $\{K_{j1}, K_{j2}\}$ to the S_j over a reliable channel.

Step 3 After obtaining $\{K_{j1}, K_{j2}\}$ from the RC, S_j keeps $\{K_{j1}, K_{j2}\}$ as a secret parameter and declares that the server's identity SID_j is known to the legitimate user U_i only.

5.2 User Registration Phase

For registering a new user with the system, the following steps are performed.

Step 1 Firstly, the U_i imprints biometric f_i and inputs the identity ID_i and password PW_i.

Step 2 The U_i picks up a random number K and computes $CPW_i = h(PW_i \parallel ID_i \parallel K)$, $B_i = H(f_i \parallel K)$. Then the U_i puts forward the information $\{ID_i, CPW_i, B_i\}$ to the RC through a trustworthy channel and also submits his own secret credentials such as passport and driving license number to RC through a trustworthy channel.

Step 3 The RC checks the user's identity ID_i in his database. If ID_i already exists in the database, then the RC asks U_i to select another ID_i. The RC inspects the registration information of the U_i also, and if the U_i is a new user, the RC sets $N = 0$; otherwise, $N = N + 1$.

Step 4 The RC computes $UID_i = h(ID_i \parallel N \parallel x)$, $A_i = h(ID_i \parallel x)$, $C_i = A_i \oplus h(CPW_i \parallel B_i)$, $D_i = y_s \oplus h(ID_i \parallel CPW_i)$ and $E_i = h(A_i \parallel CPW_i \parallel B_i)$. After that, the RC stores UID_i ID_i, N and U_i's secret credentials in the database.

Step 5 The RC sends a smart card to the U_i that contains $\{UID_i, C_i, D_i, E_i, h(.), H(.)\}$, where y_s is a secret key shared between the registration center and the legal user U_i.

Table 2 Login and authentication phase of the proposed scheme

User U_i	Registration center RC	Server S_j
Insert SC into terminal and imprints f_i and keys ID_i and PW_i		
Computes $K = KN \oplus h(ID_i \parallel PW_i)$, $CPW_i = h(ID_i \parallel PW_i \parallel K)$, $B_i = H(f_i \parallel K)$, $A_i = C_i \oplus h(CPW_i \parallel B_i)$, $y_s = D_i \oplus h(ID_i \parallel CPW_i)$ and $E_i' = h(A_i \parallel CPW_i \parallel B_i)$		
If $(E_i' \neq E_i)$, Reject		
Else generates random nonce R_1 and Computes $M_1 = E_{h(A_i \parallel y_s)}(SID_j \parallel R_1)$, $M_2 = B_i \oplus h(ID_i \parallel R_1 \parallel y_s \parallel T_1)$ and $M_3 = h(UID_i \parallel B_i \parallel R_1 \parallel SID_j \parallel T_1)$.		
$\xrightarrow[unreliablechannel]{\{UID_i, M_1, M_2, M_3, T_1\}}$		
	If$(T_2 - T_1 \leq \triangle T)$ is false, Reject	
	Else the RC retrieves ID_i from the database and computes $A_i = h(ID_i \parallel x)$. Then the RC decrypts the message M_1, i.e., $(SID_j \parallel R_1)$= $D_{h(A_i \parallel y_s)}(M_1)$ and computes $B_i = M_2 \oplus h(ID_i \parallel R_1 \parallel y_s \parallel T_1)$ and $M_3' = h(UID_i \parallel B_i \parallel R_1 \parallel SID_j \parallel T_1)$.	
	If $M_3' \neq M_3$, quits the session.	
	Else RC retrieves N_j from the database and computes $K_{j1} = h(SID_j \parallel x)$ and $K_{j2} = h(K_{j1} \parallel N_j \parallel y_s)$, $M_4 = h(SID_j \parallel K_{j1}) \oplus R_1$, $M_5 = h(K_{j2} \parallel SID_j) \oplus R_2$, $M_6 = (ID_i \parallel B_i) \oplus h(R_1 \parallel R_2 \parallel SID_j \parallel T_3)$ and $M_7 = h(ID_i \parallel B_i \parallel R_1 \parallel R_2 \parallel T_3)$, where R_2 is random nonce generated by RC.	
	$\xrightarrow[unreliablechannel]{\{M_4, M_5, M_6, M_7, T_3\}}$	If($T_4 - T_3 \leq \triangle T$) is false, Reject
		Else computes $R_1 = M_4 \oplus h(SID_j \parallel K_{j1})$, $R_2 = M_5 \oplus h(K_{j2} \parallel SID_j)$, $(ID_i \parallel B_i) = M_6 \oplus h(R_1 \parallel R_2 \parallel SID_j \parallel T_3)$ and M_7'=$h(ID_i \parallel B_i \parallel R_1 \parallel R_2 \parallel T_3)$.
		If $(M_7 \neq M_7)$, Reject
		Else creates a random nonce R_3 and calculates $M_8 = h(K_{j2}) \oplus h(R_1 \parallel ID_i)$, $M_9 = h(h(K_{j2}) \parallel R_1 \parallel B_i \parallel T_5) \oplus R_2$, $M_{10} = h(R_2 \parallel h(K_{j2}) \parallel SID_j) \oplus R_3$ and $M_{11} = h(T_5 \parallel R_2 \parallel R_3 \parallel B_i \parallel ID_i)$. Finally the S_j sends $\{M_8, M_9, M_{10}, M_{11}, T_5\}$ to the U_i
		$\xleftarrow[unreliablechannel]{\{M_8, M_9, M_{10}, M_{11}, T_5\}}$
If($T_6 - T_5 \leq \triangle T$) is not hold, terminates the session		
Else The U_i computes $h(K_{j2}) = M_8 \oplus h(R_1 \parallel ID_i)$, $R_2 = M_9 \oplus h(h(K_{j2}) \parallel R_1 \parallel B_i \parallel T_5)$, $R_3 = M_{10} \oplus h(R_2 \parallel h(K_{j2}) \parallel SID_j)$ and M_{11}'=$h(T_5 \parallel R_2 \parallel R_3 \parallel B_i \parallel ID_i)$		
If $(M_{11}' \neq M_{11})$, Reject		
Else computes $SK = h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$ and $M_{12} = h(SK \parallel B_i \parallel T_7)$. U_i sends the message $\{M_{12}, T_7\}$ to the S_j.		
$\xrightarrow[unreliablechannel]{\{M_{12}, T_7\}}$		If($T_8 - T_7 \leq \triangle T$) is not hold, terminates the session
		Else S_j computes SK'=$h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$, M_{12}'=$h(SK' \parallel B_i \parallel T_7)$ and matches M_{12}'=M_{12}. If this is true, then mutual authentication holds and U_i and S_j agree upon a common session key SK.

Step 6 After getting the smart card from the RC, U_i computes $KN = K \oplus h(ID_i \parallel PW_i)$ and securely stores KN into the smart card. Finally, the smart card holds $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$.

5.3 Login Phase

Whenever the U_i wants the services of the remote server, then the following steps are performed.

Step 1 The U_i inserts the smart card into the terminal and imprints f_i into specific device attached with the system. Then, the U_i inputs ID_i and PW_i.

Step 2 The smart card computes $K = KN \oplus h(ID_i \parallel PW_i)$, $CPW_i = h(ID_i \parallel PW_i \parallel K)$, $B_i = H(f_i \parallel K)$, $A_i = C_i \oplus h(CPW_i \parallel B_i)$, $y_s = D_i \oplus h(ID_i \parallel CPW_i)$ and $E'_i = h(A_i \parallel CPW_i \parallel B_i)$. It checks if $E'_i = E_i$. If this is true, then the user U_i is assumed to be a legitimate user.

Step 3 After checking the originality of the user, the smart card takes a random nonce R_1 and computes $M_1 = E_{h(A_i \| y_s)}(SID_j \parallel R_1)$, $M_2 = B_i \oplus h(ID_i \parallel R_1 \parallel y_s \parallel T_1)$ and $M_3 = h(UID_i \parallel B_i \parallel R_1 \parallel SID_j \parallel T_1)$, where T_1 is the current time stamp.

Step 4 Finally, the U_i sends the message $\{UID_i, M_1, M_2, M_3, T_1\}$ to the RC over an untrustworthy channel.

5.4 Authentication Phase

After getting the message $\{UID_i, M_1, M_2, M_3, T_1\}$ from the U_i, the server S_j and the registration center RC execute the following steps.

Step 1 Upon receiving the login message $\{UID_i, M_1, M_2, M_3, T_1\}$ at time T_2, the RC checks the condition $T_2 - T_1 \leq \triangle T$, where $\triangle T$ is maximum transmission delay. If this condition is not true, then RC rejects the U_i's login request message; otherwise, RC performs the next step.

Step 2 RC retrieves ID_i corresponding to UID_i from the database and computes $A_i = h(ID_i \parallel x)$. Then the RC decrypts the message M_1 and retrieves SID_j and R_1, i.e., $(SID_j \parallel R_1) = D_{h(A_i \| y_s)}(M_1)$.

Step 3 RC computes $B_i = M_2 \oplus h(ID_i \parallel R_1 \parallel y_s \parallel T_1)$ and $M'_3 = h(UID_i \parallel B_i \parallel R_1 \parallel SID_j \parallel T_1)$. After that, RC checks if $M'_3 = M_3$. If it is true, the user U_i is assumed to be a legal one; otherwise, the session is terminated.

Step 4 The RC retrieves N_j corresponding to SID_j from the database and computes $K_{j1} = h(SID_j \parallel x)$ and $K_{j2} = h(K_{j1} \parallel N_j \parallel y_s)$. Then after the RC creates a random nonce R_2 and computes $M_4 = h(SID_j \parallel K_{j1}) \oplus R_1$, $M_5 = h(K_{j2} \parallel SID_j) \oplus R_2$, $M_6 = (ID_i \parallel B_i) \oplus h(R_1 \parallel R_2 \parallel SID_j \parallel T_3)$ and $M_7 = h(ID_i \parallel B_i \parallel R_1 \parallel R_2 \parallel T_3)$. Then, the RC sends the message $\{M_4, M_5, M_6, M_7, T_3\}$ to the server S_j over an unreliable channel.

Step 5 After getting the message from the RC, the server first checks the condition $T_4 - T_3 \leq \triangle T$, where T_4 is current time stamp and $\triangle T$ is maximum transmission delay. If this condition is false, the session is terminated; otherwise, S_j executes next step.

Step 6 The server computes $R_1 = M_4 \oplus h(SID_j \parallel K_{j1})$, $R_2 = M_5 \oplus h(K_{j2} \parallel SID_j)$, $(ID_i \parallel B_i) = M_6 \oplus h(R_1 \parallel R_2 \parallel SID_j \parallel T_3)$ and $M'_7 = h(ID_i \parallel B_i \parallel R_1 \parallel R_2 \parallel T_3)$. After that, the S_j compares $M'_7 = M_7$. If this condition holds, the S_j trusts the legitimacy of the RC and executes the next step; otherwise, it terminates the session.

Step 7 The S_j creates a random nonce R_3 and calculates $M_8 = h(K_{j2}) \oplus h(R_1 \parallel ID_i)$, $M_9 = h(h(K_{j2}) \parallel R_1 \parallel B_i \parallel T_5) \oplus R_2$, $M_{10} = h(R_2 \parallel h(K_{j2}) \parallel SID_j) \oplus R_3$ and $M_{11} = h(T_5 \parallel R_2 \parallel R_3 \parallel B_i \parallel ID_i)$. Then, the S_j provides the message $\{M_8, M_9, M_{10}, M_{11}, T_5\}$ to the user.

Step 8 After obtaining the message from the S_j at current time stamp T_6, the user first verifies $T_6 - T_5 \leq \triangle T$; if it is true, then U_i performs the next step; otherwise, it terminates the session.

Step 9 The user U_i computes $h(K_{j2}) = M_8 \oplus h(R_1 \parallel ID_i)$, $R_2 = M_9 \oplus h(h(K_{j2}) \parallel R_1 \parallel B_i \parallel T_5)$, $R_3 = M_{10} \oplus h(R_2 \parallel h(K_{j2}) \parallel SID_j)$ and $M'_{11} = h(T_5 \parallel R_2 \parallel R_3 \parallel B_i \parallel ID_i)$ and matches $M'_{11} = M_{11}$. If it holds, then the U_i trusts the originality of the server and computes $SK = h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$ and $M_{12} = h(SK \parallel B_i \parallel T_7)$. Finally, U_i provides the message $\{M_{12}, T_7\}$ to the server S_j.

Step 10 Upon getting the message from U_i, the server S_j checks the validity of the time stamp, i.e., $T_8 - T_7 \leq \triangle T$, where T_8 is current time stamp and $\triangle T$ is maximum transmission delay. The server S_j computes $SK' = h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$, $M'_{12} = h(SK' \parallel B_i \parallel T_7)$ and matches $M'_{12} = M_{12}$. If it is true, then mutual authentication holds and U_i and S_j agree upon a common session key SK.

5.5 Password and Biometric Update Phase

Whenever the U_i wants to update password and biometric, the following steps are performed.

Step 1 The U_i inserts his smart card into the terminal and enters ID_i, PW_i and imprints f_i.

Step 2 The smart card calculates $K = KN \oplus h(ID_i \parallel PW_i)$, $CPW_i = h(ID_i \parallel PW_i \parallel K)$, $B_i = H(f_i \parallel K)$, $A_i = C_i \oplus h(CPW_i \parallel B_i)$, $y_s = D_i \oplus h(ID_i \parallel CPW_i)$ and $E'_i = h(A_i \parallel CPW_i \parallel B_i)$. It verifies if $E'_i = E_i$; if true, it means that the U_i is a legal user and then the smart card reader asks to enter new password PW_i^{new} and biometric f_i^{new}; otherwise, it expires the session.

Step 3 Smart card reader computes $CPW_i^{new} = h(ID_i \parallel PW_i^{new} \parallel K)$, $B_i^{new} = H(f_i^{new} \parallel K)$, $C_i^{new} =$

$C_i \oplus h(CPW_i \parallel B_i) \oplus h(CPW_i^{new} \parallel B_i^{new})$, $D_i^{new} = D_i \oplus h(ID_i \parallel CPW_i) \oplus h(ID_i \parallel CPW_i^{new})$, $KN^{new} = KN \oplus h(ID_i \parallel PW_i) \oplus h(ID_i \parallel PW_i^{new})$ and $E_i^{new} = h(C_i \oplus h(CPW_i \parallel B_i) \parallel CPW_i^{new} \parallel B_i^{new})$. Then, the smart card replaces the old values of $\{C_i, D_i, E_i\ KN\}$ with the new values $\{C_i^{new}, D_i^{new}, E_i^{new}, KN^{new}\}$.

5.6 Smart Card Revocation Phase

When the smart card of a user is lost or stolen, then it should be revoked. For revoking a smart card, the following steps are performed.

Step 1 The user U_i submits his/her secret credentials such as passport number and driving license number to the *RC* through the secure channel.

Step 2 The *RC* checks the secret credentials provided by the user U_i is correct or not. If these are incorrect, *RC* rejects the request; otherwise, it performs the next step.

Step 3 The value of N is incremented by one for each revocation request by the *RC*. The user U_i re-registers with the *RC* without altering his/her ID_i. Here, the user U_i is strongly suggested not to use any previous values for re-registration; otherwise, someone who got the smart card may fabricate the user U_i by using the previously stored parameters in the lost or stolen smart card.

6 Authentication Proof Based on *BAN* Logic

In this segment, we validate our proposed protocol with the help of *BAN* logic. The *BAN* logic is used for analyzing the authenticated protocols and ensures that the scheme achieves the session key agreement and mutual authentication securely [36]. There are some basic rules and notations of *BAN* logic as given in [10]. On the basis of these rules, we perform the following steps for verifying the presented scheme.

Step 1 We consider six goals of the proposed scheme as follows.

Goal1: $RC| \equiv (RC \overset{SK}{\longleftrightarrow} U_i)$
Goal2: $RC| \equiv U_i| \equiv (RC \overset{SK}{\longleftrightarrow} U_i)$
Goal3: $S_j| \equiv (S_j \overset{SK}{\longleftrightarrow} RC)$
Goal4: $S_j| \equiv RC| \equiv (S_j \overset{SK}{\longleftrightarrow} RC)$
Goal5: $S_j| \equiv (S_j \overset{SK}{\longleftrightarrow} U_i)$
Goal6: $S_j| \equiv U_i| \equiv (S_j \overset{SK}{\longleftrightarrow} U_i)$

Step 2 We transform the proposed scheme into idealized form as follows.

Message1: $UID_i, M_2, M_3, T_1, M_1 :< R_1 >_{h(A_i \| y_s)}$
Message2: $M_4, M_6, M_7, T_3, M_5 :< R_2 >_{h(K_{j2} \| SID_j)}$
Message3: $M_8, M_9, M_{10}, M_{11} :< R_3 >_{ID_i}$

Step 3 The nine assumptions are considered as follows for further analysis.

A1: $U_i| \equiv \sharp\{R_1, R_2, R_3\}$
A2: $S_j| \equiv \sharp\{R_1, R_2, R_3\}$
A3: $RC| \equiv \sharp\{R_1, R_2, R_3\}$
A4: $RC| \equiv RC \overset{h(A_i \| y_s)}{\longleftrightarrow} U_i$
A5: $S_j| \equiv S_j \overset{h(K_{j2} \| SID_j)}{\longleftrightarrow} RC$
A6: $U_i| \equiv U_i \overset{ID_i}{\longleftrightarrow} S_j$
A7: $RC| \equiv U_i \Rightarrow R_1$
A8: $S_j| \equiv RC \Rightarrow R_2$
A9: $U_i| \equiv S_j \Rightarrow R_3$

Step 4 On the basis of nine assumptions and fundamental rules of *BAN* logic [10], we prove the accuracy of the proposed protocol as follows.

By using the Message1, we can write
S1: $RC \triangleleft \{UID_i, M_2, M_3, T_1, M_1 :< R_1 >_{h(A_i \| y_s)}\}$
By using the Message Meaning Rule, the assumption A4 and S1, we can acquire
S2: $RC| \equiv U_i| \sim \{R_1\}$
With the help Nonce Verification Rule, S2 and assumption A3, we can obtain
S3: $RC| \equiv U_i| \equiv \{R_1\}$
With the help of Jurisdiction Rule, S3, assumption A7, we can get
S4: $RC| \equiv \{R_1\}$, where R_1 is the prominent parameter in the session key
According to S3, assumption A3 and Session Key Rule, we obtain
S5: $RC| \equiv (RC \overset{SK}{\longleftrightarrow} U_i)$ **Goal1 is proved.**
By using Nonce Verification Rule, the assumption A3 and S5, we could get
S6: $RC| \equiv U_i| \equiv (RC \overset{SK}{\longleftrightarrow} U_i)$ **Goal2 is proved.**
From the Message2, we could write
S7: $S_j \triangleleft \{M_4, M_6, M_7, T_3, M_5 :< R_2 >_{h(K_{j2} \| SID_j)}\}$
We could get from S7, assumption A5 and Message Meaning Rule
S8: $S_j| \equiv RC| \sim \{R_2\}$
From S8, assumption A2 and Nonce Verification Rule, we can obtain
S9: $S_j| \equiv RC| \equiv R_2$
With the help of assumption A8, S9 and Jurisdiction Rule, we could achieve
S10: $S_j| \equiv R_2$
From assumption A2, Session Key Rule and S9, we gain
S11: $S_j| \equiv (S_j \overset{SK}{\longleftrightarrow} RC)$ **Goal3 is proved.**

By using Nonce Verification Rule, A2 and S11, we achieve

S12: $S_j | \equiv RC | \equiv (S_j \overset{SK}{\longleftrightarrow} RC)$ **Goal4 is proved.**

By using the Message3, we can write

S13: $U_i \triangleleft \{M_8, M_9, M_{10}, M_{11} :< R_3 >_{ID_i}\}$

According to Message Meaning Rule, S13 and A6, we acquire

S14: $U_i | \equiv S_j | \sim \{R_3\}$

From assumption A1, Nonce Verification Rule and S14, we obtain

S15: $U_i | \equiv S_j | \equiv R_3$

According to Jurisdiction Rule, S15 and A9, we achieve

S16: $U_i | \equiv R_3$

Using Session Key Rule, assumption A1 and S15, we get

S17: $S_j | \equiv (S_j \overset{SK}{\longleftrightarrow} U_i)$ **Goal5 is proved**

According to S17, Nonce Verification Rule and A1, we obtain

S18: $S_j | \equiv U_i | \equiv (S_j \overset{SK}{\longleftrightarrow} U_i)$ **Goal6 is proved**

The above proof shows that both user and server believe the session key securely shared between themselves.

7 Informal Security Analysis

This section scrutinizes the security of the presented scheme against various security threats.

Proposition 1 *The presented scheme defends from the identity and password guessing attack.*

Proof Assume that user U_i uses easily memorable ID_i and PW_i, which can be guessed in polynomial time as per the Threat model. Thus, $\mathcal{A}$ tries to guess ID_i or PW_i using the extracting parameters $\{UID_i, C_i, D_i, E_i, KN\}$ from the smart card's memory and the communicating messages $\{UID_i, M_1, M_2, M_6, M_7, M_8, M_{11}\}$. However, $\mathcal{A}$ cannot obtain the U_i's ID_i and PW_i as discussed below:

1. Suppose that $\mathcal{A}$ gets the UID_i, C_i, D_i, E_i and KN from the smart card, where $UID_i = h(ID_i \parallel N \parallel x)$, $A_i = h(ID_i \parallel x)$, $C_i = A_i \oplus h(CPW_i \parallel B_i)$, $D_i = y_s \oplus h(ID_i \parallel CPW_i)$, $E_i = h(A_i \parallel CPW_i \parallel B_i)$, CPW_i=h$(PW_i \parallel ID_i \parallel K)$ and B_i=H$(f_i \parallel K)$. We can observe that the parameter C_i is protected with the help of hash function; therefore, $\mathcal{A}$ cannot obtain $\{ID_i, PW_i\}$ from the parameter C_i. If $\mathcal{A}$ tries to guess ID_i and PW_i, the guessing probability would be approximately $\frac{1}{2^{12n+160+1024}}$, where the length of x is 1024 bit and length of K is 160 bit. It may be noted that guessing U_i's biometric is an arduous task [33] and hence $\mathcal{A}$ cannot guess f_i. Therefore, $\mathcal{A}$ cannot get ID_i and PW_i from the parameter C_i.
2. The value D_i relies on $\{y_s, ID_i, PW_i, K\}$ and was protected by using the hash function. So, $\mathcal{A}$ cannot extract $\{ID_i, PW_i\}$ from D_i. Furthermore, if $\mathcal{A}$ attempts to guess ID_i and PW_i, $\mathcal{A}$ has to know four unknown values $\{y_s, ID_i, PW_i, K\}$ at the same time and their guessing probability would be $\frac{1}{2^{12n+160+1024}}$, which is negligible.
3. The value of E_i relies on $\{ID_i, x, PW_i, f_i, K\}$, which is secured due to the hash function; thus, $\mathcal{A}$ was unable to derive ID_i and PW_i. Additionally $\mathcal{A}$ attempts to guess ID_i and PW_i, and for this, $\mathcal{A}$ requires $\{ID_i, PW_i, f_i, K, x\}$ at the same time and their guessing probability is $\frac{1}{2^{12n+160+1024}}$, which is negligible.
4. The parameter UID_i relies on $\{ID_i, N, x\}$ and was secured by the hash function. So, $\mathcal{A}$ cannot derive ID_i. Next, if $\mathcal{A}$ chooses a guessed ID_i and tries to verify it, he/she needs to guess three unknown values $\{x, N, ID_i\}$ at the same time which is not feasible. The guessing probability would be approximately $\frac{1}{2^{6n+160+1024}}$, where the length of N is 160 bits.
5. During the execution of the login phase, $\mathcal{A}$ intercepts the message $\{UID_i, M_1, M_2, M_3\}$, where $UID_i = h(ID_i \parallel N \parallel x)$, $M_1 = E_{h(A_i \parallel y_s)}(SID_j \parallel R_1)$ and $M_2 = B_i \oplus h(ID_i \parallel R_1 \parallel y_s \parallel T_1)$. An attacker tries to retrieve ID_i from the parameter M_1. For this $\mathcal{A}$ requires A_i and y_s at the same time, where y_s is a secret key shared between U_i and RC and $A_i = h(ID_i \parallel x)$. But $\mathcal{A}$ does not know the A_i and y_s, and without knowing these values, $\mathcal{A}$ cannot derive ID_i.
6. The parameter M_2 depends on $\{B_i, ID_i, R_1, y_s, T_1\}$ and secured due to the hash function. So, $\mathcal{A}$ cannot derive ID_i from M_2. Moreover, an adversary $\mathcal{A}$ tries to guess ID_i by using M_2 parameter, whose probability would be $\frac{1}{2^{6n+160+1024}}$. It may be noted that the guessing of U_i's biometric is an arduous task [33], making $\mathcal{A}$ difficult to guess f_i. Therefore, $\mathcal{A}$ cannot get ID_i from the parameter M_2.
7. During the execution of authentication phase, $\mathcal{A}$ traps the communication message $\{M_6, M_7, M_8, M_{11}\}$, where $M_6 = (ID_i \parallel B_i) \oplus h(R_1 \parallel R_2 \parallel SID_j \parallel T_3)$, $M_7 = h(ID_i \parallel B_i \parallel R_1 \parallel R_2 \parallel T_3)$ $M_8 = h(K_{j2}) \oplus h(R_1 \parallel ID_i)$ and $M_{11} = h(T_5 \parallel R_2 \parallel R_3 \parallel B_i \parallel ID_i)$. The parameter M_6 is based on $\{ID_i, B_i, R_1, R_2, SID_j\}$, and if $\mathcal{A}$ tries to obtain ID_i from M_6, he/she requires four unknown values $\{B_i, R_1, R_2, SID_j\}$ at the same time. Moreover, if $\mathcal{A}$ attempts to guess IDi_i by utilizing the parameter M_6, the guessing probability would be $\frac{1}{2^{6n+160+160+160}}$.
8. The parameter M_7 is reliable due to the h(.), $\mathcal{A}$ cannot get ID_i from the M_7. Additionally, $\mathcal{A}$ requires four parameters $\{ID_i, R_1, R_3, B_i\}$ at the same time to guess the ID_i, whose probability is equivalent to $\frac{1}{2^{6n+160+160}}$.

9. If the $\mathcal{A}$ tries to procure ID_i from the M_8, $\mathcal{A}$ has to guess three unknown values $\{ID_i, R_1, K_{j2}\}$ at the same time, which is not feasible. The guessing probability would be approximately $\frac{1}{2^{6n+160+160}}$.
10. The parameter M_{11} is protected by utilizing the hash function, so $\mathcal{A}$ is not capable of extracting ID_i from M_{11}. If $\mathcal{A}$ wants to guess the ID_i, he/she has to guess the values $\{ID_i, R_2, R_3, B_i\}$ at the same time, which is not possible in polynomial time and the guessing probability of ID_i is negligible.

The above discussion affirms that $\mathcal{A}$ cannot obtain the U_i's $\{ID_i, PW_i\}$ and the probability of guessing this confidential information is negligible. Therefore, the presented scheme defends from the password and identity guessing attacks.

Proposition 2 *The presented scheme defends from the user and server impersonation attack.*

Proof Suppose that an adversary $\mathcal{A}$ traps the message and then tries to create another fake login or reply message using the extracted smart card parameters and intercepted communication message. $\mathcal{A}$ cannot behave as U_i or S_j as discussed below:

1. $\mathcal{A}$ traps the login message $\{UID_i, M_1, M_2, M_3, T_1\}$ and tries to compute new forged message, where $UID_i = h(ID_i \parallel N \parallel x)$, $M_1 = E_{h(A_i \parallel y_s)}(SID_j \parallel R_1)$, $M_2 = B_i \oplus h(ID_i \parallel R_1 \parallel y_s \parallel T_1)$ and $M_3 = h(UID_i \parallel B_i \parallel R_1 \parallel SID_j \parallel T_1)$. To compute the login message M_1, $\mathcal{A}$ requires $\{R_1, ID_i, x, y_s, SID_j\}$, where R_1 is random nonce, ID_i is the user's identity, SID_j is server's identity, and x and y_s are secret keys. Though attacker can generate R_1, yet four values $\{ID_i, x, y_s, SID_j\}$ are still unknown to attacker, and in Proposition 1, we have proved that $\mathcal{A}$ cannot obtain or guess the ID_i. Therefore, $\mathcal{A}$ cannot creates M_2 parameter without knowing $\{ID_i, x, y_s, SID_j\}$.
2. In order to compute the parameter M_2, $\mathcal{A}$ needs four values $\{B_i, R_1, ID_i, y_s, T_1\}$, where B_i is U_i's biometric and T_1 is current time stamp. We know that $\mathcal{A}$ can generate the random nonce R_1 and the time stamp T_1. But $\mathcal{A}$ cannot obtain the $\{ID_i, B_i, y_s\}$ using the smart card parameters and intercepted login message, which is already discussed in Proposition 1, and without knowing $\{ID_i, B_i, y_s\}$, he/she cannot compute the forged message M_2.
3. If $\mathcal{A}$ wants to imitate the message M_3, $\mathcal{A}$ needs four parameters $\{UID_i, SID_j, R_1, B_i, T_1\}$. Note that the $\mathcal{A}$ can produce random nonce R_1 and current time stamp T_1 and $\mathcal{A}$ also knows the parameter UID_i from the previously intercepted login message. $\mathcal{A}$ still does not know the SID_j and B_i. Thus, $\mathcal{A}$ cannot generate the value M_3 without knowing SID_j and B_i.
4. Next, if $\mathcal{A}$ wants to impersonate as server, he/she tries to create the reply message $\{M_8, M_9, M_{10}, M_{11}, T_5\}$, where $M_8 = h(K_{j2}) \oplus h(R_1 \parallel ID_i)$, $M_9 = h(h(K_{j2}) \parallel R_1 \parallel B_i \parallel T_5) \oplus R_2$, $M_{10} = h(R_2 \parallel h(K_{j2}) \parallel SID_j) \oplus R_3$ and $M_{11} = h(T_5 \parallel R_2 \parallel R_3 \parallel B_i \parallel ID_i)$. To compute the reply message M_8, $\mathcal{A}$ requires $\{K_{j2}, R_1, ID_i\}$, but $\mathcal{A}$ does not know these values. In Proposition 1 we have already shown that the $\mathcal{A}$ cannot retrieve ID_i; therefore, $\mathcal{A}$ cannot compute M_8.
5. If $\mathcal{A}$ tries to compute the parameter M_9, he/she requires $\{K_{j2}, R_1, B_i, T_5, R_2\}$. It is to note that $\mathcal{A}$ knows only the current time stamp T_5 and rest of other parameters $\{K_{j2}, R_1, B_i, R_2\}$ are unrevealed to $\mathcal{A}$. Thus $\mathcal{A}$ cannot generate the reply message $\{M_9\}$
6. If $\mathcal{A}$ wants to compute the forged message M_{10}, he/she requires the parameters $\{R_2, K_{j2}, R_3, SID_j\}$. It may be noted that the $\mathcal{A}$ can generate the random nonce R_3, but three parameters are still unknown to $\mathcal{A}$, i.e., SID_j, R_2 and K_{j2}, and without knowledge of these values he/she cannot compute the forged message $\{M_{10}\}$.
7. If $\mathcal{A}$ wants to imitate the message M_{11}, $\mathcal{A}$ needs four parameters $\{T_5, R_2, R_3, B_i, ID_i\}$. We know that the $\mathcal{A}$ can easily generate random nonce R_3 and the time stamp T_5. However, rest of other parameters $\{R_2, B_i, ID_i\}$ are unknown to $\mathcal{A}$, and in Proposition 1, we have already proved that $\mathcal{A}$ cannot obtain ID_i. Therefore, $\mathcal{A}$ is unable to compute M_{11} without knowing these parameters.

The above discussion shows that the presented protocol defends from the user and server spoofing attack.

Proposition 3 *The presented protocol defends from the lost/stolen smart card attack.*

Proof We suppose that $\mathcal{A}$ has got the U_i's smart card and got all the secret information $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$ from it. However, the presented protocol protects from the lost/stolen smart card attack as justified below:

1. $\mathcal{A}$ tries to obtain or guess the ID_i and PW_i by utilizing the smart card extracted values $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$. But, $\mathcal{A}$ cannot obtain or guess $\{ID_i, PW_i\}$ from the extracted values of the smart card, as has already been discussed in Proposition 1.
2. $\mathcal{A}$ wants to act as U_i or S_j with the help of obtained smart card parameters $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$. But, we have already known from the Proposition 2 that $\mathcal{A}$ cannot behave as U_i or S_j using the smart card information $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$.

The above discussions affirm that the proposed protocol defends from the lost/stolen smart card attack.

Proposition 4 *The presented protocol protects from the smart card theft attack.*

Proof In authentication, the smart card theft attack is very crucial. In this attack, $\mathcal{A}$ wants to produce a new smart card with the help of own confidential information like PW_i and f_i and without modifying the real U_i's ID_i and S_j's information. Our scheme protects from the smart card theft attack as follows:

1. Let $\mathcal{A}$ gets the U_i's smart card and extracts all the confidential values $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$ from it, where $UID_i = h(ID_i \parallel N \parallel x)$, $A_i = h(ID_i \parallel x)$, $C_i = A_i \oplus h(CPW_i \parallel B_i)$, $D_i = y_s \oplus h(ID_i \parallel CPW_i)$, $E_i = h(A_i \parallel CPW_i \parallel B_i)$, CPW_i=h($PW_i \parallel ID_i \parallel K$), B_i=H($f_i \parallel K$) and $KN = K \oplus h(ID_i \parallel PW_i)$
2. $\mathcal{A}$ easily computes B_a= H($f_a \parallel K_a$) using his/her own biometric f_a and newly generated random number K_a.
3. Next $\mathcal{A}$ tries to compute A_i that relies on $\{ID_i, x\}$. In Proposition 1, we have already shown that $\mathcal{A}$ cannot retrieve ID_i using the extracted values of the smart card and the communication messages. Thus, without knowing ID_i and x, the computation of A_i is not possible.
4. Similarly, $\mathcal{A}$ tries to compute the parameter CPW_i that depends on ID_i, PW_i and K. But the $\mathcal{A}$ has no way to obtains or guess the U_i's ID_i, and without the knowledge of ID_i, $\mathcal{A}$ cannot implant new PW_a. Therefore, the computation of CPW_i is not possible, and without the knowledge of CPW_i and A_i, the attacker cannot compute C_i, D_i, and E_i. Thus, $\mathcal{A}$ cannot issue a new smart card without the knowledge of A_i, C_i, D_i and E_i.

The above justification shows that the proposed protocol protects from smart card theft attack.

Proposition 5 *The presented protocol provides the perfect forward secrecy.*

Proof The forward secrecy defined as if the secret keys are exposed and an attacker $\mathcal{A}$ tries to calculate the session SK with the help of these exposed secret keys. But $\mathcal{A}$ could not succeed to compromise the past or future session key. The presented protocol facilitates the forward secrecy property as follows:

1. The session key $SK = h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$, where $\{R_1, R_2, R_3\}$ are random nonces, ID_i and SID_j are the identities of user and server, respectively; the parameter $h(K_{j2})$ is securely shared between legitimate U_i and S_j.
2. Suppose that $\mathcal{A}$ obtains the x and y_s by some means and then tries to compute A_i to decrypt the message M_1 in order to retrieve $\{SID_j, R_1\}$, i.e., $(SID_j \parallel R_1)$= $D_{h(A_i \parallel y_s)}(M_1)$. But $\mathcal{A}$ requires ID_i to compute A_i and in Proposition 1, we have already delineated that the attacker has no way to retrieve ID_i. Thus, an attacker cannot compute A_i; as a result, he/she cannot obtain $\{SID_j, R_1\}$, which is the essential parameter for the computation of the session key SK.
3. Next, $\mathcal{A}$ tries to compute K_{j1} and K_{j2} by utilizing the exposed secret key x and y_s, where $K_{j1} = h(SID_j \parallel x)$, $K_{j2} = h(K_{j1} \parallel N_j \parallel y_s)$, SID_j is server's identity, and N_j is random nonce. But, $\mathcal{A}$ cannot compute K_{j1} without the knowledge of SID_j because SID_i is kept secret. It means that the SID_j is not stored in smart card and also not directly sent along with the communication message. As a result, the $\mathcal{A}$ cannot compute K_{j2}, which is relied on K_{j1} and random nonce N_j. Therefore, $\mathcal{A}$ cannot obtain R_1, R_2, ID_i and R_3 without the awareness of SID_j, K_{j1} and K_{j2}.
4. Thus, $\mathcal{A}$ cannot compute SK without the knowledge of confidential values $\{R_1, R_2, R_3, ID_i, h(K_{j2})\}$. Therefore in the presented scheme if the secret key x and y_s is exposed by some means, then from this exposure, the secrecy of the SK does not get affected.

Thus, the presented protocol facilitates the forward secrecy property.

Proposition 6 *The presented protocol defends from the known session-specific temporary information attack.*

Proof Let $\mathcal{A}$ has got the ephemeral secret key and tries to compute the SK, which are derived from using short-term keys. The following steps show that the scheme protects from the known session-specific temporary information threat.

1. In the presented protocol, the $SK = h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$, where $\{R_1, R_2, R_3\}$ are short-term keys and $\{ID_i, SID_j, h(K_{j2})\}$ are confidential parameters.
2. Assume that $\mathcal{A}$ has obtained the ephemeral keys $\{R_1, R_2, R_3\}$ and tries to compute the SK. But, $\mathcal{A}$ fails to compute session key SK because the session key not only depends on the short-term key, but also on secret information $\{ID_i, SID_j, h(K_{j2})\}$. In the presented scheme, the parameters SID_j and $h(K_{j2})$ are kept secret, known only legitimate U_i. Additionally, $\mathcal{A}$ cannot derive ID_i, as discussed in Proposition 1.
3. Therefore, $\mathcal{A}$ was unable to calculate the session key without the awareness of ID_i, SID_j and $h(K_{j2})$ even if the short-term keys $\{R_1, R_2, R_3\}$ are exposed to $\mathcal{A}$.

Thus, the presented protocol defends from the known session-specific temporary information threat.

Proposition 7 *The presented protocol prevents from the privileged insider attack.*

Proof The insider attack is a very influential attack in a remote user authenticated scheme. In recent times, the bulk of the remote user authentication protocols are breakable because of the insider attack. In this attack, the wicked insider attempts to acquire the secret information of the user, such as password to access the other account of the user. The proposed scheme prevents the insider attack as follows:

1. In the presented protocol, the user put forwards the registration message $\{ID_i, CPW_i, B_i\}$ to the RC, where $CPW_i = \text{h}(PW_i \parallel ID_i \parallel K)$ and $B_i = \text{H}(f_i \parallel K)$.
2. The insider attacker cannot obtain PW_i of the user because of non-invertible hash function.
3. Moreover, if the insider attacker guesses PW_i^* and inspects the PW_i^* is accurate or not, he/she needs to guess two unknown parameters $\{PW_i, K\}$ at the same time, which is infeasible.
4. The guessing probability to obtain the PW_i from the parameter CPW_i is $\frac{1}{2^{6n+160}}$, which is negligible.

Thus, the presented protocol holds up the privileged insider attack.

Proposition 8 *The presented protocol defends from the replay attack.*

Proof In replay attack, adversary $\mathcal{A}$ taps the communication message, and after some time, he/she replays it and tries to prove that the message is transmitted from the legitimate entity. In the proposed scheme if an adversary $\mathcal{A}$ taps the communication message and after some time $\mathcal{A}$ again sends the message to the recipient entity. Upon obtaining the message, the recipient entity detects that the received message is the replicated message and was send by the attacker due to the freshness of time stamp T and verification of the transmission delay $\triangle T$. Therefore, the proposed protocol always rejects the attacker's replicated message due to illegal transmission delay time. However, time stamp-based remote authentication schemes suffer from clock synchronization problem. We have assumed that the protocol maintains global clock for synchronizing the time stamp. Thus, the presented protocol is able to defend from the replay attack.

Proposition 9 *The presented protocol provides efficient login and password change phase.*

Proof During the login phase, the smart card reader first verifies the legitimacy of the U_i, i.e., $E_i' = E_i$, where $E_i = h(A_i \parallel CPW_i \parallel B_i)$, $A_i = h(ID_i \parallel x)$, $CPW_i = h(PW_i \parallel ID_i \parallel K)$ and $B_i = \text{H}(f_i \parallel K)$. If this is true, then only smart card generates the login message $\{UID_i, M_1, M_2, M_3, T_1\}$ and transmits to the RC. Therefore, if the registered/illegal user inputs incorrect ID_i, PW_i and f_i to log into the server, it is detected by the smart card reader promptly, which reduces extra computation and communication cost. Similarly, in the password change phase, the smart card ratifies the legality of the U_i, i.e., $E_i' = E_i$ before upgrading the PW_i and f_i. The PW_i and f_i are upgraded only when the genuine U_i inputs accurate information such as ID_i, PW_i and f_i. Note that in the presented protocol, the U_i upgrades the f_i and PW_i without taking help from RC, which minimize the communication and computation cost.

Proposition 10 *The presented protocol achieves known key security property.*

Proof Known key security defined as if previously established SK is revealed to $\mathcal{A}$ by some means; then from this revelation, the secrecy of past or future SK should not be influenced. In the proposed protocol, the $SK = h(SID_j \parallel ID_i \parallel h(K_{j2}) \parallel R_1 \parallel R_2 \parallel R_3)$, where $\{R_1, R_2, R_3\}$ are random nonces, ID_i is U_i's identity, SID_j is S_j's identity, and $h(K_{j2})$ is a secret key shared between legal U_i and S_j. In our scheme, the $\mathcal{A}$ cannot extract any parameters from the SK due to the hash function. Moreover, the random nonces $\{R_1, R_2, R_3\}$ are changed in each authentication session; as a result, the session key is also changed in each authentication session. Therefore, $\mathcal{A}$ was unable to compute past or future SK based on previously exposed SK.

8 Security Affirmation Using *AVISPA* Tool

In this section, we simulate the presented protocol using *AVISPA* (Automated Validation of Internet Security Protocols and Applications) tool [37,38]. The *AVISPA* tool has four backend models, namely *OFMC*, $CL-AtSe$, *SATMAC* and *TA4SP*. For detailed description of the *AVISPA* tool, refer [5,7,22,33]. We have implemented code in *HLPSL* (High-Level Protocol Specification Language) for the U_i, S_j and RC including session and environment which are described in the subsection specifying the proposed protocol. Moreover, the simulation results assert that the presented protocol protects from the passive and active attacks and also can defend against the man-in-the-middle and replay attacks.

8.1 Specifying the Proposed Protocol

This section shows the role of user U_i, registration center RC, server S_j and session and environment in *HLPSL*. We have presented the key role of U_i in the Fig. 1. Initially, U_i transmits the registration request message $\{ID_i, CPW_i, B_i\}$ to the registration center RC over the genuine channel using the symmetric key SK_{ur} and $Snd()$ operation. The type of declaration channel(dy) specifies that it follows the Dolev Yao [39] attack model. The declaration secret($\{PW_i, F_i\}$, *subs*1, U_i) points out that only U_i knows the confidential value $\{PW_i, F_i\}$. Finally, U_i gains a smart card holding the information $\{UID_i, C_i, D_i, E_i\}$ through a trustworthy channel using symmetric key SK_{ur} and receives operation *Rcv*(). During the execution of login phase, U_i produces a random nonce $R1$ and time stamp T_1 using new operation and computes the login message $\{UID_i, M_1, M_2, M_3, T_1\}$. After that U_i transmits the message $\{UID_i, M_1, M_2, M_3, T_1\}$ to the registration center through the unreliable channel. The declaration secret(R_1', *subs*2, $\{U_i, RC, S_j\}$) shows that the random nonce R_1' is only known to the U_i, S_j and RC. The declaration witness(U_i, RC, *user_regcentre_r*1, R_1') states that the U_i generates R_1' for the RC. In the authentication phase, the U_i receives a message $\{M_8, M_9, M_{10}, M_{11}, T_5\}$ from the S_j through unreliable channel. Lastly, the U_i sends the message $\{M_{12}, T_7\}$ to the S_j over untrustworthy channel. The declaration secret(K_{j2}, *subs3*, $\{U_i, RC, S_j\}$) indicates

```
role user(Ui,Sj,RC: agent,
        SKur: symmetric_key,
         SKsr: symmetric_key,
         H: hash_func,
             Snd,Rcv:channel(dy))
played_by Ui
def=
local State:nat,
CPWi,PWi,IDi,K,Fi,Bi,N,X,Ai,Ci,Di,Ei,KN,Ys,SIDj,R1,R2,R3,SK,Kj2,Kj1,Nj:text,
UIDi,M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12,T1,T3,T5,T7:message,
Inc:hash_func
const user_server_r1, user_regcentre_r1,server_user_r3,regcentre_user_r2,
subs1,subs2,subs4,subs5,subs6,subs7,subs8:protocol_id
init State:=0
transition
1.State=0/\Rcv(start)=|>
State':=1/\K':=new()
/\CPWi':=H(PWi.IDi.K')
/\Bi':=H(Fi.K')
%Send registration request message
/\Snd({IDi.CPWi'.Bi'}_SKur)
/\secret({PWi,Fi},subs1,Ui)
2.State=1/\Rcv({UIDi.Ci.Di.Ei}_SKur)=|>
State':=2/\R1':=new()
/\T1':=new()
/\KN':=xor(K,H(IDi.PWi))
/\M1':={SIDj.R1'}_(H(Ai.Ys))
/\M2':=xor(Bi,H(IDi.R1'.Ys.T1'))
/\M3':=H(UIDi.Bi.R1'.SIDj.T1')
/\Snd(UIDi.M1'.M2'.M3'.T1')
/\secret({R1'},subs2,{Ui,RC,Sj})
/\witness(Ui,RC,user_regcentre_r1,R1')
3.State=2/\Rcv(M8.M9.M10.M11.T5)=|>
State':=3/\T7':=new()
/\Kj2':=xor(M8,H(R1.IDi))
/\R2':=xor(M9,H(Kj2.R1.Bi.T5))
/\R3':=xor(M10,H(R2.Kj2'.SIDj))
/\SK':=H(SIDj.IDi.Kj2.R1.R2.R3)
/\M12':=H(SK'.Bi.T7')
/\Snd(M12.T7')
/\secret({Kj2'},subs3,{Ui,Sj,RC})
/\request(Sj,Ui,server_user_r3,R3)
/\request(RC,Ui,regcentre_user_r2,R2)
end role
```

Fig. 1 The key role of the U_i in HLPSL

```
role regcentre(Ui,Sj,RC: agent,
        SKur: symmetric_key,
         SKsr: symmetric_key,
         H: hash_func,
             Snd,Rcv:channel(dy))
played_by RC
def=
local State:nat,
CPWi,PWi,IDi,K,Fi,Bi,N,X,Ai,Ci,Di,Ei,KN,Ys,SIDj,R1,R2,KRS,R3,SK,Kj2,Kj1,Nj:text,
UIDi,M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12,T1,T3,T5,T7:message,
Inc:hash_func
const user_server_r1, user_regcentre_r1,server_user_r3,regcentre_user_r2,
subs1,subs2,subs3,subs4,subs5,subs6,subs7,subs8:protocol_id
init State:=0
transition
1. State=0/\Rcv({SIDj}_SKsr)=|>
State':=1/\Nj':=new()
/\Kj1':=H(SIDj.X)
/\Kj2':=H(Kj1.Nj.Ys)
/\Snd({Kj1.Kj2}_SKsr)
/\secret({Kj1},subs4,{Sj,RC})
/\secret({Ys},subs5,{Ui,RC})
/\secret({X},subs6,RC)
2.State=1/\Rcv({IDi.CPWi.Bi}_SKur)=|>
State':=2/\UIDi':=H(IDi.N.X)
/\Ai':=H(IDi.X)
/\Ci':=xor(Ai,H(CPWi.Bi))
/\Di':=xor(Ys,H(IDi.CPWi))
/\Ei':=H(Ai.CPWi.Bi)
/\Snd({UIDi'.Ci'.Di'.Ei'}_SKur)
3.State=2/\Rcv(UIDi.M1.M2.M3.T1)=|>
State':=3/\R2':=new()
/\T3':=new()
/\M4':=xor(R1,H(SIDj.Kj1))
/\M5':=xor(R2,H(Kj2,SIDj))
/\M6':=xor(H(IDi.Bi),H(R1.R2'.SIDj.T3'))
/\M7':=H(IDi.Bi.R1.R2'.T3')
/\Snd(M4'.M5'.M6'.M7'.T3')
/\secret({R2'},subs7,{RC,Ui,Sj})
/\request(Ui,RC,user_regcentre_r1,R1)
end role
```

Fig. 2 The key role of the *RC* in HLPSL

that only U_i, RC and S_j know the secret key K_{j2}. The declaration request($\{RC, U_i\}$, $regcentre\_user\_r2$, R_2) tells that the RC validates the U_i. The declaration request(S_j, U_i, $server\_user\_r3$, R_3) shows that the server S_j authenticates the U_i.

In Fig. 2, the main role of the registration center in the HLPSL has been divulged. Initially, in the registration phase, registration center receives SID_j from S_j over the trustworthy channel using $Rcv()$ operation and SK_{sr}. The RC generates a random nonce N_j and computes K_{j1} and K_{j2}. After that, RC transmits K_{j1} and K_{j2} to the S_j over a secure channel. The declaration secret(Y_s, $subs5$, $\{U_i, RC\}$) specifies that the Y_s was only revealed to RC and U_i. The declaration secret (K_{j1}, $subs4$, $\{S_j, RC\}$) divulges that only S_j and RC know the K_{j1}. The declaration secret (X, $subs6$, RC) tells that the X is kept secret to RC only. The RC receives a registration message $\{ID_i, CPW_i, B_i\}$ from the U_i using the symmetric key SK_{ur}. Finally, the RC provides a smart card carrying the secret information $\{UID_i, C_i, D_i, E_i\}$ to the U_i. During the authentication phase, the RC gets message $\{UID_i, M_1, M_2, M_3, T_1\}$ from the U_i using $Rcv()$ operation through the open channel. At last, the RC

```
role server(Ui,Sj,RC: agent,
        SKur: symmetric_key,
         SKsr: symmetric_key,
         H: hash_func,
        Snd,Rcv:channel(dy))
played_by Sj
def=
local State:nat,
CPWi,PWi,IDi,K,Fi,Bi,N,X,Ai,Ci,Di,Ei,KN,Ys,SIDj,R1,R2,R3,SK,Kj2,Kj1,Nj:text,
UIDi,M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12,T1,T3,T5,T7:message,
Inc:hash_func
const user_server_r1, user_regcentre_r1,server_user_r3,regcentre_user_r2,
subs1,subs2,subs3,subs4,subs5,subs6,subs7,subs8:protocol_id
init State:=0
transition
1.State=0/\Rcv(start)=|>
State':=1/\SIDj':=new()
/\Snd({SIDj'}_SKsr)
2.State=1/\Rcv({Kj1.Kj2}_SKsr.M4.M5.M6.M7.T3)=|>
State':=2/\R3':=new()
/\T5':=new()
/\M8':=xor(Kj2,H(R1.IDi))
/\M9':=xor(R2,H(Kj2.R1.Bi.T5))
/\M10':=xor(R3',H(R2.Kj2.SIDj))
/\M11':=H(T5.R2.R3.Bi.IDi)
/\Snd(M8'.M9'.M10'.M11'.T5')
/\request(Ui,Sj,user_server_r1,R1)
/\request(RC,Sj,regcentre_server_r2,R2)
/\witness(Sj,Ui,server_user_r3,R3')
/\secret({R3'},subs8,{Sj,Ui})
3.State=2/\Rcv(M12.T7)=|>
State':=3/\SK':=H(SIDj.IDi.Kj2.R1.R2.R3)
/\M12':=H(SK'.Bi.T7)
end role
```

Fig. 3 The key role of the S_j in HLPSL

produces random nonce R_2 and dispatches $\{M_4, M_5, M_6, M_7, T_3\}$ to the S_j via unreliable channel. The declaration secret(R_2', *subs*7, $\{RC, U_i, S_j\}$) points out that only *RC*, S_j and U_i know the random nonce R_2. Additionally, the declaration request (U_i, *RC*, *user_regcentre_r*1, R_1) indicates the strong authentication property of the *RC* by U_i on R_1, i.e., *RC* requests U_i to authenticate the random nonce R_1.

Figure 3 shows the key role of the S_j in the *HLPSL* specification. Firstly, the S_j generates SID_j using the *new*() operation and transmits SID_j to the *RC* via reliable channel. Next, the S_j acquires the message $\{M_4, M_5, M_6, M_7, T_3\}$ from the *RC* using *Rcv*() operation. Lastly, the S_j produces random nonce R_3 and transmits $\{M_8, M_9, M_{10}, M_{11}, T_5\}$ to the U_i via unfaithful channel. The declaration request ($\{U_i, S_j\}$, *user_server_r*1, R_1) denotes the S_j requests to the U_i for verifying R_1. Next, the declaration witness(S_j, U_i, *server_user_r*3, R_3) designates that the S_j creates the random nonce R_3 for the U_i. Moreover, the declaration

```
role session(Ui,Sj,RC: agent,
        SKur: symmetric_key,
         SKsr: symmetric_key,
         H: hash_func)
def=
local SI,SJ,RI,RJ,TI,TJ:channel(dy)
composition
user(Ui,Sj,RC,SKur,SKsr,H,SI,RI)
/\server(Ui,Sj,RC,SKur,SKsr,H,SJ,RJ)
/\regcentre(Ui,Sj,RC,SKur,SKsr,H,TI,TJ)
end role
role environment()
def=
const ui,sj,rc:agent,
skur:symmetric_key,
sksr:symmetric_key,
h:hash_func,
cpwi,pwi,idi,k,fi,bi,n,x,ai,ci,di,uidi,ei,kn,ys,t1,sidj,r1,r2,r3,sk,kj2,kj1,nj:text,
 user_server_r1, user_regcentre_r1,server_user_r3,regcentre_user_r2,
subs1,subs2,subs3,subs4,subs5,subs6,subs7,subs8:protocol_id
intruder_knowledge={ui,sj,rc,h,uidi.ci.di.ei}
composition
session(ui,sj,rc,skur,sksr,h)
/\session(ui,sj,rc,skur,sksr,h)
/\session(sj,ui,rc,skur,sksr,h)
end role
goal
secrecy_of subs1
secrecy_of subs2
secrecy_of subs3
secrecy_of subs4
secrecy_of subs5
secrecy_of subs6
secrecy_of subs7
secrecy_of subs8
authentication_on user_server_r1
authentication_on regcentre_user_r2
authentication_on server_user_r3
end goal
environment()
```

Fig. 4 The key role of the session and environment in HLPSL

secret(R_3, $subs8$, $\{S_j, U_i\}$) delineates that the R_3 is revealed to the S_j and U_i only. The declaration request($\{RC, S_j\}$, $regcentre\_server\_r2$, R_2) denotes the S_j requests to the RC for verifying R_2. Lastly, S_j receives message $\{M_{12}, T_7\}$ from the U_i over unreliable channel.

Figure 4 provides the key role of session and environment in *HLPSL* specification. Three authentications and eight secrecy goals are discussed as follows.

– $secrecy\_of\ subs1$: It states that the secret value $\{PW_i, F_i\}$ is only known to the legitimate user U_i.
– $secrecy\_of\ subs2$: It expresses that only the U_i, RC and S_j are kept R_1 as a secret.
– $secrecy\_of\ subs3$: It tells that K_{j2} is only known to U_i, RC and S_j.
– $secrecy\_of\ subs4$: It says that only RC and S_j know the secret value K_{j1}.

- *secrecy_of subs*5 : It indicates that the Y_s is secret shared between U_i and the RC.
- *secrecy_of subs*6: It says that the X is only known to the RC.
- *secrecy_of subs*7: It points out that R_2 is shared between U_i, S_j and RC only.
- *secrecy_of subs*8:It states that R_3 is shared between U_i and S_j only.
- *authentication_on user_server_r*$_1$: It represents that the U_i creates the random nonce R_1, where only U_i knows the random nonce R_1. When the S_j gets R_1 from U_i, then the S_j authenticates U_i.
- *authentication_on regcentre_user_r*$_2$: It represents that the RC produces the random nonce R_2, where R_2 is only known to RC. When the U_i receives R_2 from the RC, the U_i legitimates RC.
- *authentication_on server_user_r*3: It denotes that the S_j produces the random nonce R_3, where R_3 is kept secret to S_j only. When the U_i receives R_3 from S_j, the U_i validates S_j.

8.2 Simulation Results

In this segment, we present the simulation results of the presented protocol using broadly known *AVISPA* tool. Figures 5 and 6 contain the simulation results using the *OFMC* and *CL − AtSe* backend models, which assert that the proposed scheme is *SAFE*. As a result, the presented protocol defends from passive and active attacks and was also safe from the man-in-the-middle and replay threats.

```
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
/home/preeti/documents/span/testsuite/results/ijcs.if
GOAL
as_specified
BACKEND
OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime: 1.10s
visitedNodes: 64 nodes
depth: 6 plies
```

Fig. 5 The output of OFMC

```
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
TYPED_MODEL
PROTOCOL
/home/preeti/documents/span/testsuite/results/ijcs.if
GOAL
As Specified
BACKEND
CL-AtSe
STATISTICS
Analysed   : 0 states
Reachable  : 0 states
Translation: 0.29 seconds
Computation: 0.00 seconds
```

Fig. 6 The output of CL-AtSe

9 Formal Security Analysis

In this part, the formal security verification of the proposed protocol using the random oracle model is presented and certifies that the proposed protocol is more reliable. We follow the formal security analysis of the proposed scheme, alike in [5,7]. For understanding the notion of the random oracle model, we use the formal definition and then prove the theorems as given below.

Definition 1 *The* $h(k)$ *is considered as a negligible function if for every* $d > 0$, $\exists$ *an integer* k_0 *such that* $h(k) < k^{-d}$, *for every* $k \geq k_0$.

Definition 2 *Collision resistant is defined as* $Adv_{\mathcal{A}}^{H}(t) = prb(m, m') \Longleftarrow_R \mathcal{A}$ *and* $h(m) = h(m')$, *where* $prb[m, m']$ *denotes the probability of an event* (m, m') *in a random experiment,* $\Longleftarrow_R \mathcal{A}$ *denotes that the pair of messages* (m, m') *is chosen by* $\mathcal{A}$, *and* $Adv_{A}^{H}(t)$ *denotes the probability's advantages over random choice by* $\mathcal{A}$ *for the period of time* t. *The hash function* $h(.)$ *is called as collision resistant if* $Adv_{A}^{H}(t) \leq \epsilon$, *for any small positive values* $\epsilon > 0$.

Definition 3 *This is considered as an oracle when it provides hash input* m *without any condition from the comparable hash output* y, *where* $y = h(m)$, *it is called* $RORACLE()$.

Algorithm 1 $Exp1^{HASH}_{\mathcal{A},ITAS}$

Input: $\langle UID_i, C_i, D_i, E_i, KN, h(.), H(.), M_1, M_2, M_3, T_1\rangle$
Output: 1 or 0.
Call Reveal oracle on E_i to retrieve the information A_i, CPW_i and B_i as $(A_i' \parallel CPW_i' \parallel B_i') \leftarrow Reveal(E_i)$
Computes $C_i' = A_i' \oplus h(CPW_i' \parallel B_i')$
if $(C_i' == C_i)$ **then**
  Call Reveal oracle on input CPW_i' for obtaining the information ID_i, PW_i and K as $(PW_i^* \parallel ID_i^* \parallel K^*) \leftarrow Reveal(CPW_i')$
  Computes $KN^* = K^* \oplus h(ID_i^* \parallel PW_i^*)$
  if $(KN^* == KN)$ **then**
    Accept ID_i^* and PW_i^* is the accurate identity and password of the U_i
    Call Reveal oracle on M_3 for getting the information UID_i, B_i, R_1, SID_j and T_1 as $(UID_i^{**} \parallel B_i^{**} \parallel R_1^{**} \parallel SID_j^{**} \parallel T_1^{**}) \leftarrow Reveal(M_3)$
    if $(UID_i^{**} == UID_i$ and $T_1^{**} == T_1)$ **then**
      Call Reveal oracle on B_i^{**} for getting the information f_i and K as $(f_i^{***} \parallel K^{***}) \leftarrow Reveal(B_i^{**})$
      Computes $B_i^{***} = H(f_i^{***} \parallel K^{***})$
      Computes $M_3' = h(UID_i \parallel B_i^{***} \parallel R_1^{**} \parallel SID_j^{**} \parallel T_1)$
      if $(M_3' == M_3)$ **then**
        Accept f_i^{***} as exact biometric of the U_i
        Return1 Success
      else
        Return0 Failure
      end if
    else
      Return0 Failure
    end if
  else
    Return0 Failure
  end if
else
  Return0 Failure
end if

Theorem 1 *Suppose a non-invertible hash function behaves as a random oracle and an attacker knows all the smart card's parameters {UID_i, C_i, D_i, E_i, KN, $h(.)$, $H(.)$} and communication messages {UID_i, M_1, M_2, M_3, T_1}, $\mathcal{A}$ was still unable to extract ID_i, PW_i and f_i of the U_i.*

Proof We first consider that $\mathcal{A}$ has capability to retrieve identity ID_i, password PW_i and biometric f_i of the U_i. Let an attacker gets the smart card of U_i and extracts all the secret values $\{UID_i, C_i, D_i, E_i, KN, h(.), H(.)\}$ with the help of power consumption analysis [34,35]. Also, $\mathcal{A}$ intercepts the login message $\{UID_i, M_1, M_2, M_3, T_1\}$ and runs the experimental algorithm $Exp1^{HASH}_{\mathcal{A},ITAS}$ for retrieving the values ID_i, PW_i and f_i. The success probability for $Exp1^{HASH}_{\mathcal{A},ITAS}$ is defined by $Success1 = |Pr[Exp1^{HASH}_{\mathcal{A},ITAS} = 1] - 1|$, where Pr(.) denotes the probability of $Exp1^{HASH}_{\mathcal{A},ITAS}$. Then the advantage function for algorithm $Exp1^{HASH}_{\mathcal{A},ITAS}$ is defined as: $Adv1(t_1, q_{r1}) = Max_{\mathcal{A}}\{Succss1\}$, where the maximum is based on the execution time et_1 and number of queries q_{r1} made to the oracle Reveal. The proposed scheme is considered as provably safe from $\mathcal{A}$ for retrieving ID_i, PW_i and f_i of the U_i if $Adv1(t_1, q_{r1}) \leq \epsilon$ for any positive small value $\epsilon > 0$. From the $Exp1^{HASH}_{\mathcal{A},ITAS}$, if the attacker has the capability to reverse the non-invertible hash function, then only $\mathcal{A}$ can retrieve the ID_i, PW_i and f_i and win the game. However, the input derived from the non-invertible hash function is a computationally infeasible task. So, $Adv1(t_1, q_{r1}) \leq \epsilon$ for any positive small value $\epsilon > 0$. Therefore, the proposed scheme protects against $\mathcal{A}$ for obtaining the U_i's ID_i, PW_i and f_i.

Algorithm 2 $Exp2^{HASH}_{\mathcal{A},ITAS}$

Input: $\langle UID_i, C_i, D_i, E_i, KN, h(.), H(.), M_{12}, T_7\rangle$
Output: 1 or 0.
Call Reveal oracle on input UID_i to retrieve the information ID_i, N, x as $(ID_i' \parallel N' \parallel x') \leftarrow Reveal(UID_i)$
Computes $UID_i^* = h(ID_i' \parallel N' \parallel x')$
if $(UID_i^* == UID_i)$ **then**
  Accept x' as the accurate secret key of the RC.
  Call Reveal oracle on M_{12} for acquiring the information SK, B_i and T_7 as $(SK^* \parallel B_i^* \parallel T_7^*) \leftarrow Reveal(M_{12})$
  if $(T_7^* == T_7)$ **then**
    Accept SK^* as the correct session key
    Return1 Success
  else
    Return0 Failure
  end if
else
  Return0 Failure
end if

Theorem 2 *Let a non-invertible hash function behaves as a random oracle and $\mathcal{A}$ knows all the parameters {UID_i, C_i, D_i, E_i, KN, $h(.)$, $H(.)$} of the smart card and communication message {M_{12}, T_7} via public channel, $\mathcal{A}$ still cannot obtain x and SK.*

Proof The proof of this theorem is same as the Theorem 1. We consider that $\mathcal{A}$ has the capability to retrieve x and SK. An attacker $\mathcal{A}$ can extract the confidential information $\{UID_i$, C_i, D_i, E_i, KN, $h(.)$, $H(.)\}$ from the smart card's memory and obstruct the communication message $\{M_{12}, T_7\}$. An attacker $\mathcal{A}$ executes $Exp2^{HASH}_{\mathcal{A},ITAS}$ algorithm for retrieving the SK and x. The success probability for $Exp2^{HASH}_{\mathcal{A},ITAS}$ is defined by $Success2 = |Pr[Exp2^{HASH}_{\mathcal{A},ITAS} = 1] - 1|$, where Pr(.) denotes the probability of $Exp2^{HASH}_{\mathcal{A},ITAS}$. Then the advantage function for algorithm $Exp2^{HASH}_{\mathcal{A},ITAS}$ is defined as: $Adv2(t_2, q_{r2}) = Max_{\mathcal{A}}\{Success2\}$, where the maximum is based on the execution time et_2 and number of queries q_{r2} made to the oracle Reveal. The presented scheme is considered provably safe from $\mathcal{A}$ for retrieving secret key x and SK, if $Adv2(t_2, q_{r2}) \leq \epsilon$ for any positive value $\epsilon > 0$. According to the $Exp2^{HASH}_{\mathcal{A},ITAS}$, if the attacker has the capability to reverse the non-invertible hash function, then only $\mathcal{A}$ can easily retrieve the SK and x and win the game. However, the input derived from the hash function is a computationally infeasible task. So $Adv2(t_2, q_{r2}) \leq \epsilon$ for any small $\epsilon > 0$. Therefore, the proposed scheme is able to defend against $\mathcal{A}$ for retrieving the SK and x.

10 Performance Evaluation

In this section, we discuss the performance evaluation of the proposed protocol and compare it with that of the methods [9,11–14,23,24,30] in terms of the security features, communication cost, computation cost and estimated execution time.

10.1 The Comparison of Computation Cost and Estimated Time

Table 3 contains the communication cost, computation cost and estimated time (in seconds) of presented scheme along with other related authentication schemes [9,11–14,23,24,30]. Here, we have used the following notations: T_H: non-invertible hash function, T_S: the symmetric key encryption/decryption operation and T_E: the operation of modular exponentiation. From Table 3, we can observe that the computation cost of the proposed protocol is better than other protocols [9,11,12,30]. The protocols [13,14,23,24] have slightly better performance than that of the proposed scheme, but these protocols suffer from various security attacks and also do not provide all security attributes as discussed later in subsection Security Features. Thus, the scheme [13,14,23,24] is not applicable for real-life applications because of various types of security threats. For computing the execution time of the presented protocol and other relevant protocols, we have assumed that the hash function takes 0.0005 seconds, the symmetric key encryption/decryption operation takes 0.0087 seconds, and the modular exponentiation operation takes 0.522 seconds [4,40]. The estimated time for execution of the relevant schemes [9,11–14,23,24,30] and the proposed scheme is 0.0658, 3.2096, 5.816, 0.019, 0.0205, 0.0284, 0.0165, 0.097 and 0. 0479 seconds, respectively.

10.2 The Comparison of Security Features

In Table 4, we have delineated the comparison of security features of our scheme with other relevant schemes [9,11–14,23,24,30]. From this table, we identify that the schemes [9,12,14,23,24,30] do not provide user anonymity property and the schemes [9,12,14] are also not secure against the password guessing attack. Further, the protocols [9,14,23,24] do not resist the user and server impersonation attack, and the protocols [11–13] are not secured against the replay attack, insider attack and session key temporary information attack. From Table 4, we observe that no protocols under discussion are secured against various security barriers and also do not provide all security attributes. In contrast, the proposed scheme defends against all the security attacks and also provides numerous security features.

10.3 The Communication Cost Comparison

Table 3 and Fig. 7 contain the communication cost of the proposed scheme along with related schemes [9,11–14,23,24,30]. For computing the communication cost, we have supposed that the length of ID_i, PW_i, random nonce,

Table 3 Performance Comparison: communication cost, Computation cost, Estimated time

Schemes ⇒	Ref [11]	Ref [12]	Ref [13]	Ref [14]	Ref [23]	Ref [24]	Ref [9]	Ref [30]	Proposed
CC	3904	5856	960	1280	1152	1600	3232	4032	3072
CCRP	$4T_H$	$3T_H$	$13T_H$	$10T_H$	$4T_H$	$5T_H$	$9T_H+2T_S$	$4T_H$	$9T_H$
CCLP	$3T_H+2T_E+1T_S$	$1T_H+2T_S+1T_E$	$9T_H$	$10T_H$	$3T_H+1T_S$	$6T_H$	$8T_H+1T_S$	$3T_H+1T_S$	$8T_H+1T_S$
CCAP	$7T_H+7T_S+4T_E$	$9T_H+11T_S+10T_E$	$10T_H$	$14T_H$	$11T_H+1T_S$	$18T_H$	$45T_H+1T_S$	$9T_H+9T_S$	$28T_H+1T_S$
CCPH	$5T_H$	-	$6T_H$	$7T_H$	$4T_H$	$4T_H$	-	$4T_H$	$16T_H$
TCC	$16T_H+8T_S+6T_E$	$13T_H+13T_S+11T_E$	$38T_H$	$41T_H$	$22T_H+2T_S$	$33T_H$	$62T_H+4T_S$	$20T_H+10T_S$	$61T_H+2T_S$
ET	3.2096	5.8616	0.019	0.0205	0.0284	0.0165	0.0658	0.097	0.0479

PC Performance comparison, *CC* communication cost, *CCRP* computation cost of registration phase, *CCLP* computation cost of login phase, *CCAP* computation cost of authentication phase, *CCPH* computation cost of password change phase, *TCC* total computation cost and *ET* estimated time (s)

Table 4 Security Aspects Comparison

Schemes ⇒	Ref [11]	Ref [12]	Ref [13]	Ref [14]	Ref [23]	Ref [24]	Ref [9]	Ref [30]	Proposed
A1	Yes	No	Yes	No	Yes	Yes	No	Yes	Yes
A2	Yes	No	Yes	No	No	No	No	No	Yes
A3	Yes	Yes	Yes	No	No	No	No	Yes	Yes
A4	Yes	Yes	Yes	No	No	No	No	Yes	Yes
A5	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
A6	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
A7	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes
A8	Yes	No	Yes	Yes	Yes	Yes	No	No	Yes
A9	No	No	No	No	Yes	Yes	Yes	Yes	Yes
A10	Yes	-	Yes	No	No	No	No	No	Yes

A1: resist password guessing attack, A2: preserving User anonymity, A3: resist user impersonation attack, A4: resist server impersonation attack, A5: resist replay attack, A6: preserving forward secrecy property, A7: resist session key temporary information attack, A8: resist user untraceability attack, A9: resist privileged insider attack and A10: efficient login and password change phase

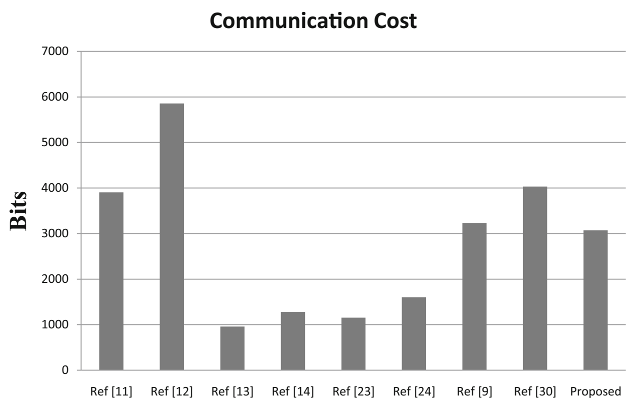

Fig. 7 Performance comparison: communication cost

time stamp and hash function (SHA-1) is of 160 bits each, the length of the symmetric encryption/decryption (AES) is 512 bits, and p, q, e and d are 1024 bits each [3]. The communication cost of the schemes [9,11–14,23,24,30] and the proposed scheme is 3232 bits, 3904 bits, 5856 bits, 960 bits, 1280 bits, 1152 bits, 1600 bits, 4032 bits and 3072 bits, respectively. Thus, the presented protocol takes less communication cost than the schemes [9,11,12,30]. In real-life application, the scheme which provides more security aspects with efficient complexity is useful.

11 Conclusion

In this article, we have cryptanalyzed Wen et al.'s protocol and found that it has various sorts of security vulnerabilities such as session key temporary information attack, inaccurate password change phase, improper authentication, the lack of smart card revocation and biometric update phase. Additionally, it does not achieve forward secrecy property. To eliminate these security weaknesses, we have discussed a new three-factor-based remote authentication protocol in multi-server environment. Using the *BAN* logic, we have shown that our scheme is safe, and also with the help of the *AVISPA* tool, we have carried out the simulation verification. We have shown using the formal and informal security verification that the proposed scheme is secured against various kinds of vulnerabilities. The performance evaluation justifies that our scheme provides better performance as compared to the existing schemes in terms of the computation cost, communication cost and execution time.

References

1. Khan, M.K.; Kumari, S.: An authentication scheme for secure access to healthcare services. J. Med. Syst. **37**(4), 1–12 (2013)
2. He, D.; Kumar, N.; Khan, M.H.; Lee, J.H.: Anonymous two-factor authentication for consumer roaming service in global mobility networks. IEEE Trans. Consum. Electron. **59**(4), 811–817 (2013)
3. Islam, S.H.; Khan, M.K.: Cryptanalysis and improvement of authentication and key agreement protocols for telecare medicine information systems. J. Med. Syst. **38**(10), 1–16 (2014)
4. Amin, R.; Biswas, G.P.: A secure three-factor user authentication and key agreement protocol for tmis with user anonymity. J. Med. Syst. **39**(8), 1–19 (2015)
5. Amin, R.; Islam, S.H.; Biswas, G.P.; Khan, M.K.; Li, X.: Cryptanalysis and enhancement of anonymity preserving remote user mutual authentication and session key agreement scheme for e-health care systems. J. Med. Syst. **39**(11), 1–21 (2015)
6. Kumari, S.; Om, H.: Authentication protocol for wireless sensor networks applications like safety monitoring in coal mines. Comput. Netw. **104**, 137–154 (2016)
7. Amin, R.; Biswas, G.P.: A novel user authentication and key agreement protocol for accessing multi-medical server usable in tmis. J. Med. Syst. **39**(3), 1–17 (2015)
8. Mishra, D.; Kumari, S.; Khan, M.K.; Mukhopadhyay, S.: An anonymous biometricbased remote userauthenticated key agreement scheme for multimedia systems. Int. J. Commun. Syst.(2015). doi:10.1002/dac.2946

9. Kim, M.; Park, N.; Won, D.: Security improvement on a dynamic ID-based remote user authentication scheme with session key agreement for multi-server environment. SecTech/CA/CES3 **339**, 122–127 (2012)
10. Amin, R.; Biswas, G.P.: Design and analysis of bilinear pairing based mutual authentication and key agreement protocol usable in multi-server environment. Wirel. Personal Commun. **84**(1), 439–462 (2015)
11. Guo, D.; Wen, F.: Analysis and improvement of a robust smart card basedauthentication scheme for multi-server architecture. Wirel. Personal Commun. **78**(1), 475–490 (2014)
12. Tsai, J.L.; Lo, N.W.; Wu, T.C.: A new password-based multi-server authentication scheme robust to password guessing attacks. Wirel. Personal Commun. **71**(3), 1977–1988 (2013)
13. Chang, C.C.; Cheng, T.F.; Hsueh, W.Y.: A robust and efficient dynamic identitybased multiserver authentication scheme using smart cards. Int. J. Commun. Syst. **29**(2), 290–306 (2016)
14. Chen, C.T.; Lee, C.C.: A two-factor authentication scheme with anonymity for multiserver environments. Secur. Commun. Netw. **8**(8), 1608–1625 (2015)
15. Tsai, J.L.; Lo, N.W.: Secure chaotic mapsbased authenticated key agreement protocol without smartcard for multiserver environments. Secur. Commun. Netw. **8**(11), 1971–1978 (2015)
16. Ford, W.; Jr.; B. S. K.: Server-assisted generation of a strong secret from a password. In: Proceedings of the 9th IEEE International Workshops on Enabling Technologies, pp. 176–180 (2000)
17. Jablon, D.P.: Password authentication using multiple servers. In: Proceedings of the RSA security conference, LNCS **2020**, 344–360 (2001)
18. Liao, Y.P.; Hsiao, C.M.: A novel multi-server remote user authentication scheme using self-certified public keys for mobile clients. Future Gener. Comput. Syst. **29**(3), 886–900 (2013)
19. Hsiang, H.C.; Shih, W.K.: Improvement of the secure dynamic ID based remote user authentication scheme for multi-server environment. Comput. Stand. Interfaces **31**(6), 1118–1123 (2009)
20. Sood, S.K.; Sarje, A.K.; Singh, K.: A secure dynamic identity based authentication protocol for multi-server architecture. J. Netw. Comput. Appl. **34**(2), 609–618 (2011)
21. Li, X.; Xiong, Y.; Ma, J.; Wang, W.: An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards. J. Netw. Comput. Appl. **35**(2), 763–769 (2012)
22. Mishra, D.; Das, A.K.; Mukhopadhyay, S.: A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards. Expert Syst. Appl. **41**(18), 8129–8143 (2014)
23. Lu, Y.; Li, L.; Peng, H.; Yang, Y.: A biometrics and smart cards-based authentication scheme for multiserver environments. Secur. Commun. Netw. **8**(17), 3219–3228 (2015)
24. Lu, Y.; Li, L.; Yang, X.; Yang, Y.: Robust biometrics based authentication and key agreement scheme for multi-server environments using smart cards. **10**(5) (2015). doi:10.1371/journal.pone.0126323 PMID:25978373
25. Chaudhry, S. A.; Naqvi, H.; Farash, M. S.; Shon, T.; Sher, M.: An improved and robust biometrics-based three factor authentication scheme for multiserver environments. J. Super Comput. 1–17 (2015). doi:10.1007/s11227-015-1601-y
26. Chaudhry, S. A.: A secure biometric based multi-server authentication scheme for social multimedia networks. Multimed. Tools Appl. **75**(20), 12705–12725 (2016). doi:10.1007/s11042-015-3194-0
27. Das, A.K.: Analysis and improvement on an efficient biometric-based remote user authentication scheme using smart cards. IET Inf. Secur. **5**(3), 145–151 (2011)
28. An, Y.: Security Analysis and Enhancements of an Effective Biometric-Based Remote User Authentication Scheme Using Smart Cards. J. Biomed. Biotechnol. **16**, 1–6 (2012). doi:10.1155/2012/519723. Article ID 519723
29. Khan, M.K.; Kumari, S.: An improved biometrics-based remote user authentication scheme with user anonymity. BioMed. Res. Int. **19** (2013). doi:10.1155/2013/491289. Article ID 491289
30. Wen, F.; Susilo, W.; Yang, G.: Analysis and improvement on a biometric-based remote user authentication scheme using smart cards. Wirel. Personal Commun. **80**(4), 1747–1760 (2015)
31. Mishra, D.; Das, A.K.; Mukhopadhyay, S.: A secure and efficient ECC-based user anonymity-preserving session initiation authentication protocol using smart card. Peer-to-Peer Netw. Appl. **9**(1), 171–192 (2016)
32. Giri, D.; Sherratt, R.S.; Maitra, T.; Amin, R.: Efficient biometric and password based mutual authentication for consumer usb mass storage devices. IEEE Trans. Consum. Electron. **61**(4), 491499 (2015)
33. Amin, R.; Biswas, G.P.; Khan, M.K.; Leng, L.; Kumar, N.: Design of an anonymity- preserving, three- factor authenticated key exchange protocol for wireless sensor networks. Comput. Netw. **101**, 42–62 (2015)
34. Kocher, P.; Jaffe, J.; Jun, B.: Differential power analysis. In: Advances in Cryptology CRYPTO 99, Lecture Notes in Computer Science. 1666, 388397 (1999)
35. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H.: Examining smart-card security under the threat of power analysis attacks. IEEE Trans. Comput. **51**(5), 541–552 (2002)
36. Burrows, M.; Abadi, M.; Needham, R.: A logic of authentication. ACM Trans. Comput. Syst. **8**(1), 1836 (1990). doi:10.1145/77648.77649
37. AVISPA: Automated validation of internet security protocols and applications. http://www.avispaproject.org/. Accessed Oct 2015
38. Tool, A. W.: http://www.avispa-project.org/web-interface/expert.php/ use on September (2015)
39. Dolev, D.; Yao, A.C.: On the security of public key protocols. IEEE Trans. Inf. Theory **29**(2), 198208 (1983)
40. Jiang, Qi; Ma, J.; Yang, G.L.L.: An efficient ticket based authentication protocol with unlinkability for wireless access networks. Wirel. Personal Commun. **77**(2), 1489–1506 (2014)