J Med Syst (2015) 39: 92
DOI 10.1007/s10916-015-0276-5

PATIENT FACING SYSTEMS

A Secure and Robust User Authenticated Key Agreement Scheme for Hierarchical Multi-medical Server Environment in TMIS

Ashok Kumar Das[1] · Vanga Odelu[2] · Adrijit Goswami[2]

Received: 26 March 2015 / Accepted: 13 July 2015 / Published online: 6 August 2015

Abstract The telecare medicine information system (TMIS) helps the patients to gain the health monitoring facility at home and access medical services over the Internet of mobile networks. Recently, Amin and Biswas presented a smart card based user authentication and key agreement security protocol usable for TMIS system using the cryptographic one-way hash function and biohashing function, and claimed that their scheme is secure against all possible attacks. Though their scheme is efficient due to usage of one-way hash function, we show that their scheme has several security pitfalls and design flaws, such as (1) it fails to protect privileged-insider attack, (2) it fails to protect strong replay attack, (3) it fails to protect strong man-in-the-middle attack, (4) it has design flaw in user registration phase, (5) it has design flaw in login phase, (6) it has design flaw in password change phase, (7) it lacks of supporting biometric update phase, and (8) it has flaws in formal security analysis. In order to withstand these security pitfalls and design flaws, we aim to propose a secure and robust user authenticated key agreement scheme for the hierarchical multi-server environment suitable in TMIS using the cryptographic one-way hash function and fuzzy extractor. Through the rigorous security analysis including the formal security analysis using the widely-accepted Burrows-Abadi-Needham (BAN) logic, the formal security analysis under the random oracle model and the informal security analysis, we show that our scheme is secure against possible known attacks. Furthermore, we simulate our scheme using the most-widely accepted and used Automated Validation of Internet Security Protocols and Applications (AVISPA) tool. The simulation results show that our scheme is also secure. Our scheme is more efficient in computation and communication as compared to Amin-Biswas's scheme and other related schemes. In addition, our scheme supports extra functionality features as compared to other related schemes. As a result, our scheme is very appropriate for practical applications in TMIS.

Keywords Telecare medicine information systems · Authentication · Key agreement · Multi-medical servers · Fuzzy extractor · Biometrics · User anonymity · AVISPA

This article is part of the Topical Collection on *Patient Facing Systems*

✉ Ashok Kumar Das
iitkgp.akdas@gmail.com; ashok.das@iiit.ac.in

Vanga Odelu
odelu.vanga@gmail.com; odelu.phd@maths.iitkgp.ernet.in

Adrijit Goswami
goswami@maths.iitkgp.ernet.in

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

[2] Department of Mathematics, Indian Institute of Technology, Kharagpur 721 302, India

Introduction

A telecare medical information system (TMIS) allows the patients to send health related information or use portals for health monitoring and healthcare-related services over the Internet or mobile networks [14]. For example, if a patient travels to a hospital, it is desirable that the expense of the patients such as travel cost and the hospitalization time is much. Thus, to reduce significantly these factors,

the patients can easily apply TMIS in order to access the healthcare delivery services.

There are several applications of TMIS including distant nursing, e-healthcare, and home monitoring facility [20, 21, 25, 28, 32, 46]. In TIMS, we have two parties: one is user from public and other is medical server, which is responsible to ensure the availability of healthcare services to the registered users via the Internet [32]. The medical server has the database of keeping records of information of its registered users including the user name, telephone number, age, address, the electronic medical record, and disclosure of any of these may endanger user's privacy. Since the telecare servers keep the electronic medical records of all registered users in the hospitals, TMIS is very useful for the physicians to make more comprehensive decision via the cooperation of some physicians in different places. However, TMIS usually works in the open environments. The security issue then becomes a crucial concern in TMIS.

In recent years, several user authentication schemes have been proposed in the literature [20, 22, 23, 26, 29, 30, 36, 39, 40, 43–45, 47, 52, 57]. In 2010, Yang and Yang [58] proposed a biometric password-based multi-server authentication scheme with smart card. However, their scheme does not resist insider attack, and has a high computational cost as it requires to perform exponential operations [42]. Sood et al. [53] then proposed a dynamic identity-based multi-server authentication scheme. Li et al. [34] pointed out that Sood et al.'s scheme fails to resist stolen verifier attack as well as stolen smart card attack. In addition, they proposed an improved smart-card based authentication scheme for multi-server architecture and it requires the involvement of a control server in order to achieve mutual authentication. Later, Wang and Ma [56] presented a smart-card based authentication scheme for multi-server environment. However, their scheme is vulnerable to privileged insider attack, server spoofing attack, impersonation attack and off-line password guessing attack [42]. Chuang and Chen [9] proposed an efficient multi-server authenticated key agreement scheme based on a user's password and biometrics. Mishra et al. [42] showed that their scheme does not resist stolen smart card attack which causes the user's impersonation attack and server spoofing attack. Mishra et al. also showed that their scheme fails to protect denial-of-service attack.

Recently, Amin and Biswas [1] presented a biometric-based authentication scheme in TMIS based on multi-medical server architecture. Though their scheme is efficient, in this paper we point out that their scheme has numerous security flaws as well as design flaws. In order to withstand those flaws found in Amin-Biswas's scheme, we aim to propose a novel and robust user authentication and key agreement scheme for the hierarchical multi-medical server architecture, which is very suitable for TMIS and secure against possible known attacks.

Threat Model

We use the Dolev-Yao threat model [19] in which any two communicating parties communicate over an insecure channel. An adversary (attacker or intruder) can eavesdrop the transmitted messages over a public insecure channel and he/she has the ability to modify, delete or change the contents of the transmitted messages. We adopt the similar threat model in our scheme in which the communicating channels are insecure and the end-point nodes (users, medical servers, physician servers in multi-medical server environment in TMIS) cannot in general be trustworthy. If a user's smart card is stolen or lost, an attacker can extract all the sensitive information stored in its memory by monitoring the power consumption of the smart card [31, 37] even if the smart card is tamper resistant.

Our contributions

Our contributions towards this paper are listed below:

- We analyze the recently proposed Amin-Biswas's scheme [1] and show that their scheme has several security loopholes as well as design flaws, such as it fails to protect privileged-insider attack, it fails to protect strong replay attack, it fails to protect strong man-in-the-middle attack, it has design flaw in user registration phase, it has design flaw in login phase, it has design flaw in password change phase, it lacks of supporting biometric update phase, and it has flaws in formal security analysis.
- We then propose a secure and robust user authenticated key agreement scheme for the hierarchical multi-server environment suitable in TMIS using the cryptographic one-way hash function and fuzzy extractor.
- Through the rigorous security analysis including the formal security analysis using the widely-accepted Burrows-Abadi-Needham (BAN) logic, the formal security analysis under the random oracle model and the informal security analysis, we show that our scheme is secure against possible known attacks.
- We simulate our scheme using the most-widely accepted and used Automated Validation of Internet Security Protocols and Applications (AVISPA) tool and the simulation results clearly show that our scheme is secure.
- Our scheme is more efficient in computation and communication as compared to Amin-Biswas's scheme and other existing related schemes.
- Our scheme also supports extra functionality features as compared to Amin-Biswas's scheme and other existing related schemes.

Organization of the Paper

The remainder of this paper is organized as follows. In Section "Useful Mathematical Preliminaries", we discuss some basic mathematical preliminaries, which are essential for describing and analyzing Amin-Biswas's scheme as well as our proposed scheme. In Section "Review of Amin-Biswas's Scheme", we briefly review different phases of their scheme, which are useful for Section "Cryptanalysis of Amin-Biswas's Scheme". In Section "The Proposed Scheme", we describe the various phases of our proposed scheme. In Section "Security Analysis of the Proposed Scheme", we show that our scheme is secure against different known attacks through the rigorous formal and informal security analysis. We simulate our scheme for the formal security verification using the widely-accepted and used AVISPA tool in Section "Simulation for Formal Security Verification using AVISPA Tool". We compare functionality features and performance of our scheme with Amin-Biswas's scheme and other related existing schemes in Section "Performance Comparison with Other Related Schemes". Finally, we conclude the paper in the next section.

Useful Mathematical Preliminaries

In this section, we briefly describe some mathematical preliminaries, which are essential for describing and analyzing Amin-Biswas's scheme [1] as well as our proposed scheme.

Collision-resistant One-way Hash Function

We define formally a one-way collision-resistant cryptographic hash function as follows [15, 51, 54].

Definition 1 (One-way collision resistant hash function) A collision-resistant one-way hash function $h : P \rightarrow Q$, where $P = \{0,1\}^*$ and $Q = \{0,1\}^n$ is a deterministic algorithm that takes an arbitrary length binary string $x \in P$ as input and produces a binary string of fixed-length n, $y \in Q$ as output. Let $Adv_{\mathcal{A}}^{HASH}(t)$ denote an adversary (attacker) $\mathcal{A}$'s advantage in finding a collision. Then,

$$Adv_{\mathcal{A}}^{HASH}(t) = Pr[(x, x') \Leftarrow_R \mathcal{A} : x \neq x' \text{ and } h(x) = h(x')],$$

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

Key Data Extraction Process from Biometric Template

We briefly describe the extraction process of key data from a given biometric of a user using a fuzzy extractor method. Note that the output of a conventional one-way hash function $h(\cdot)$ is very sensitive and it may also return completely different outputs even if there is a little variation in inputs. The biometric information is prone to various noises during data acquisition, and as a result, the reproduction of actual biometric is hard in common practice. In order to avoid such problem, a fuzzy extractor method [5, 18, 24] is preferred, which has the ability to extract a uniformly random string b_i and a public information par_i from the biometrics B_i with the error tolerance threshold t. In the reproduction process, the fuzzy extractor recovers the original biometric key data b_i for a noisy biometric B' using par_i and t. Let $\mathcal{M} = \{0,1\}^v$ be a finite v-dimensional metric space of biometric data points, $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{Z}^+$ a distance function, which can be used to calculate the distance between two points based on the metric chosen and l is the number of bits of the output string b_i, where $\mathbb{Z}^+$ denotes the set of all positive integers.

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

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

For more detailed description of the fuzzy extractor and the extraction procedure, one can refer to [5, 18].

Review of Amin-Biswas's Scheme

Amin and Biswas [1] proposed a hierarchical architecture for accessing multi-medical server system for the telecare medicine information system (TMIS), which is shown in Fig. 1. This architecture is similar to that in [8, 17] presented for hierarchical wireless sensor networks. In this architecture, they considered four types of network entities, which are discussed below:

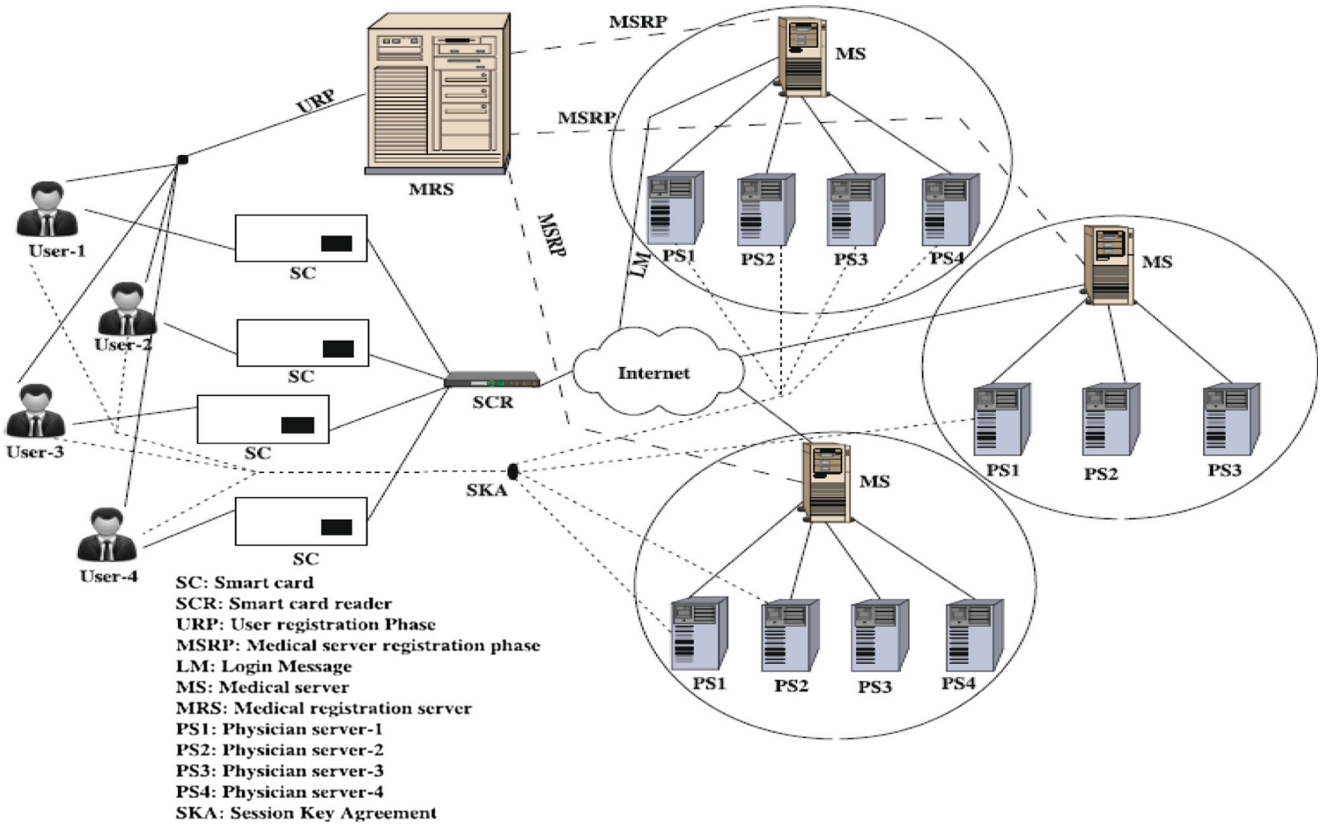

Fig. 1 Architecture for accessing multi-medical server system in Amin-Biswas's scheme (Source: [1])

- U_i (i^{th} user/patient): There are several users present in this architecture, which can access medical services from the physician servers with the help of the medical servers.
- MRS (medical registration server): There is only one MRS present in the system. MRS is responsible for providing registration to the new patients (users) U_i as well as the medical servers (MS_j), $j = 1, 2, \ldots, m$.
- MS_j (j^{th} medical server): There are several medical servers in the system.
- PS_k (k^{th} physician server): There are also several physician servers in the system.

A PS_k provides services on demand to the authorized registered users/patients U_i through a medical server MS_j. The users/patients Ui then can access PS_k through MS_j for solving personal medical related problems in TMIS. As specified in [1], PS_k may be different servers such as Cardiologist, Perinatologist, Gastroenterologist, Anesthesiologist, Hematologist, Nephrologist, Neurologist. In Amin-Biswas's scheme, there are five phases, which are described in brief in the following subsections. The notations listed in Table 1 are used for describing various phases of their scheme.

Medical Server Registration Phase

In this phase, if a medical server MS_j wants to join in the network for providing the medical services to the remote users U_i, MS_j needs to choose an identity ID_{MS_j} and send it to the MRS. After receiving ID_{MS_j}, the MRS computes the secret key $X_j = h(ID_{MS_j}||X_c)$, where X_c is the secret key of the MRS, and sends it back to MS_j via a secure channel. At the end of this phase, MS_j keeps ID_{MS_j} and X_j.

User Registration Phase

A user U_i needs to register to the MRS with the following steps:

Step 1. U_i first chooses hos/her identity ID_i, password PW_i and imprints his/her personal biometrics B_i at the sensor of a specific device. U_i then computes $PWD_i = h(ID_i||PW_i)$ and sends the message $\langle ID_i, PWD_i, B_i \rangle$ to the MRS via a secure channel.

Step 2. The MRS computes $F_i = H(B_i)$ using the biohashing function $H(\cdot)$ [27, 35], $REG_i = h(ID_i||PWD_i)$, $A_j = h(ID_i||X_j) \oplus REG_i$, and

Table 1 Notations used in this paper

Symbol	Description
U_i	i^{th} user/patient
MRS	Medical registration server
MS_j	j^{th} medical server ($1 \leq j \leq m$)
PS_k	k^{th} physician server ($1 \leq k \leq p$)
ID_i	Identity of U_i
PW_i	Password of U_i
B_i	Personal biometrics of U_i
ID_{MS_j}	Identity of MS_j
ID_k	Identity of PS_k
X_c	Secret key of the MRS
X_j	Secret key of MS_j
X_k	Shared secret key between PS_k and MS_j
R_c	Random nonce generated by U_i
R_{ms}	Random nonce generated by MS_j
R_k	Random nonce generated by PS_k
TS_c	Current time-stamp generated by U_i
TS_{ms}	Current time-stamp generated by MS_j
TS_k	Current time-stamp generated by PS_k
$\triangle T$	Maximum transmission delay or preset acceptable delay threshold or expected time interval for transmission delay or expected network delay time
$h(\cdot)$	Collision-free one-way hash function
$H(\cdot)$	Biohashing function [27, 35]
$Gen(\cdot)$	Fuzzy extractor generation algorithm
$Rep(\cdot)$	Fuzzy extractor reproduction algorithm
σ_i	Biometric key of U_i
τ_i	Biometric public parameter of U_i
t	Error tolerance threshold
$A \oplus B$	Bitwise XORed of data A with data B
$A\|\|B$	Data A concatenates with data B

$P_j = h(ID_{MS_j}|| X_j||F_i) \oplus h(REG_i||F_i)$, $1 < j \leq m$. The MRS then issues a smart card containing the information $\{\{(ID_{MS_j}, A_j, P_j)|1 < j \leq m\}, REG_i, h(\cdot), H(\cdot)\}$ and sends it to U_i via a secure channel.

Remark 1 In Amin-Biswas's scheme, after the end of the user registration phase, the smart card does not store $F_i = H(B_i)$. Also, the smart card contains the table having the tuples $\langle ID_{MS_j}, A_j, P_j\rangle$, $1 < j \leq m$. Thus, the smart card does not store the tuple $\langle ID_{MS_1}, A_1, P_1\rangle$ corresponding to the first medical server MS_1. Furthermore, it is assumed that U_i can choose very low entropy $\langle ID_i, PW_i\rangle$ which are guessable individually in polynomial time.

Login Phase

In this phase, the following steps are executed:

Step 1. A user U_i inserts his/her smart card into the card reader of a specific device, and then imprints biometric template B_i at the sensor to the specific device. The smart card computes $F_i^* = H(B_i)$. It is claimed in Amin-Biswas's scheme that the computed F_i^* is matched with the stored F_i in the smart card. However, it is seen from Remark 1 that there is no F_i stored in the smart card, and as a result, the biometric verification is not valid in Amin-Biswas's scheme. After that U_i inputs $\langle ID_i, PW_i\rangle$.

Step 2. The smart card computes $REG_i^* = h(ID_i||PW_i)$. If $REG_i^* = REG_i$, the provided $\langle ID_i, PW_i \rangle$ are valid. Otherwise, the smart card stops the session.

Step 3. Based on a medical server's identity, say ID_{MS_j}, the smart card generates a random nonce R_c and then computes $C_i = A_j \oplus REG_i$, $D_i = h(C_j||R_c)$, $E_i = P_j \oplus h(REG_i||F_i)$, $G_i = ID_i \oplus E_i$ and $L_i = E_i \oplus R_c$. The smart card then sends the message $\langle ID_{MS_j}, ID_k, F_i, D_i, G_i, L_i \rangle$ to a medical server MS_j via a public channel, where ID_k is the identity of a physician server PS_k from which U_i wants to access the medical services.

The login phase of Amin-Biswas's scheme is summarized in Table 2.

Authentication and Key Agreement Phase

In this phase, the mutual authentication and session key agreement between a user U_i and a physician server PS_k are achieved. The following steps are involved in this phase:

Step 1. After receiving the login message from U_i, MS_j computes $E_i^* = h(ID_{MS_j}||X_j||F_i)$, $ID_i^* = G_i \oplus E_i^*$, $R_c^* = L_i \oplus E_i^*$, $C_i^* = h(ID_i^*||X_j)$ and $D_i^* = h(C_i^*||R_c^*)$. After that MS_j checks the condition $D_i^* = D_i$. If it matches, MS_j believes the authenticity of U_i; otherwise, MS_j terminates the session.

Step 2. MS_j generates a random nonce R_{ms}, and computes $N_j = h(ID_k||X_k||F_i)$, $O_j = ID_i \oplus N_j$, $S_j = h(ID_i||X_k) \oplus R_{ms}$, $RAN_j = R_c^* \oplus R_{ms}$, $Q_j = h(ID_i||X_k||N_j||R_{ms})$. MS_j then sends the message $\langle ID_k, O_j, S_j, Q_j, RAN_j, F_i \rangle$ to the accessed physician server PS_k via a public channel.

Table 2 Login phase of Amin-Biswas's scheme

U_i/Smart card	MS_j
Insert the smart card and	
input ID_i, PW_i, and B_i.	
Compute $f_i^* = H(B_i)$ and check if $F_i^* = F_i$.	
Note that this condition always fails as F_i is	
not stored in the smart card.	
Compute $REG_i^* = h(ID_i||PW_i)$, and check if	
$REG_i^* = REG_i$. If it is not valid, abort the session.	
Generate R_c and compute $C_i = A_j \oplus REG_i$,	
$D_i = h(C_j||R_c)$, $E_i = P_j \oplus h(REG_i||F_i)$,	
$G_i = ID_i \oplus E_i$ and $L_i = E_i \oplus R_c$.	
$\langle ID_{MS_j}, ID_k, F_i, D_i, G_i, L_i \rangle$	
$\longrightarrow$	
(via a public channel)	

Step 3. After receiving the message in Step 2, PS_k computes $N_j' = h(ID_k||X_k||F_i)$, $ID_i' = O_j \oplus N_j'$, $R_{ms}' = h(ID_i'||X_k) \oplus S_j$, $R_c' = RAN_j \oplus R_{ms}'$, $Q_j' = h(ID_i'||X_k||N_j'||R_{ms}')$. If the condition $Q_j' = Q_j$ is satisfied, PS_k believes the authenticity of MS_j and U_i; otherwise, PS_k stops the session.

Step 4. PS_k further generates a random nonce R_k and computes a session key $SK = h(ID_i'||ID_k||R_c'||R_k)$ shared with the user U_i, $T_k = h(h(ID_i'||X_k)||SK)$, $RAN_k = R_c' \oplus R_k$, $V_k = h(ID_i'||X_k) \oplus R_k$, and sends the message $\langle T_k, RAN_k, V_k \rangle$ to U_i via a public channel.

Step 5. After receiving the message in Step 4, U_i computes $R_k^* = RAN_k \oplus R_c$, $W_k = V_k \oplus R_k^*$, $SK^* = h(ID_i||ID_k||R_c||R_k^*)$ and $T_k^* = h(W_k||SK^*)$. Finally, U_i compares T_k^* with the received T_k. If they match, U_i believes the authenticity of PS_k and also, the session key $SK^*(= SK)$ between U_i and PS_k is verified.

This phase of Amin-Biswas's scheme is summarized in Table 3.

Password Change Phase

In this phase, if a user U_i wants to change his/her old password PW_i by a desired new password PW_i^{new}, the following steps need to be executed:

Step 1. U_i first inserts his/her smart card in a specific card reader, and imprints personal biometrics B_i. After the biometric verification, U_i also inputs $\langle ID_i, PW_i \rangle$. The smart card also verifies PW_i. If it is valid, next steps are executed; otherwise, the smart card stops the phase.

Step 2. U_i inputs the new password PW_i^{new}. The smart card then computes $PWD_i^{new} = h(ID_i||PW_i^{new})$, $REG_i^{new} = h(ID_i||PWD_i^{new})$, $A_j^{new} = A_j \oplus REG_i \oplus REG_i^{new}$, $P_j^{new} = P_j \oplus h(REG_i||F_i) \oplus h(REG_i^{new}||F_i)$. The smart card then replaces $\langle REG_i, A_j, P_j \rangle$ with $\langle REG_i^{new}, A_j^{new}, B_j^{new} \rangle$, $1 < j \leq m$, respectively.

Cryptanalysis of Amin-Biswas's Scheme

In this section, we show that Amin-Biswas's scheme is vulnerable to several known attacks. In addition, we also demonstrate that their scheme has several design flaws.

Fails to Protect Privileged-insider Attack

During the user registration phase, a user U_i sends the registration request message $\langle ID_i, PWD_i, B_i \rangle$ securely to the MRS, where $PWD_i = h(ID_i||PW_i)$. Then an insider user

Table 3 Authentication and session key agreement phase of Amin-Biswas's scheme

U_i/Smart card	MS_j	PS_k
	Compute $E_i^* = h(ID_{MS_j}\|\|X_j\|\|F_i)$, $ID_i^* = G_i \oplus E_i^*$, $R_c^* = L_i \oplus E_i^*$, $C_i^* = h(ID_i^*\|\|X_j)$ and $D_i^* = h(C_i^*\|\|R_c^*)$. Check if $D_i^* = D_i$. If it does not match, terminate the session. Generate R_{ms}, and compute $N_j = h(ID_k\|\|X_k\|\|F_i)$, $O_j = ID_i \oplus N_j$, $S_j = h(ID_i\|\|X_k) \oplus R_{ms}$, $RAN_j = R_c^* \oplus R_{ms}$, $Q_j = h(ID_i\|\|X_k\|\|N_j\|\|R_{ms})$. $\langle ID_k, O_j, S_j, Q_j, RAN_j, F_i \rangle \longrightarrow$ (via a public channel)	Compute $N_j' = h(ID_k\|\|X_k\|\|F_i)$, $ID_i' = O_j \oplus N_j'$, $R_{ms}' = h(ID_i'\|\|X_k) \oplus S_j$, $R_c' = RAN_j \oplus R_{ms}'$, $Q_j' = h(ID_i'\|\|X_k\|\|N_j'\|\|R_{ms}')$. Check if $Q_j' = Q_j$. If it is not valid, stop the session. Generate R_k and compute session key $SK = h(ID_i'\|\|ID_k\|\|R_c'\|\|R_k)$, $T_k = h(h(ID_i'\|\|X_k)\|\|SK)$, $RAN_k = R_c' \oplus R_k$, $V_k = h(ID_i'\|\|X_k) \oplus R_k$.
	$\langle T_k, RAN_k, V_k \rangle \longleftarrow$	(via a public channel to U_i)
Compute $R_k^* = RAN_k \oplus R_c$, $W_k = V_k \oplus R_k^*$, $SK^* = h(ID_i\|\|ID_k\|\|R_c\|\|R_k^*)$ and $T_k^* = h(W_k\|\|SK^*)$. Check if $T_k^* = T_k$. If it is valid, U_i believes the authenticity of PS_k. Store SK^* as session key shared with PS_k.		Store SK as session key shared with U_i.

being an attacker directly knows the identity ID_i and the personal biometrics B_i. Furthermore, knowing the identity ID_i and PWD_i, the insider attacker can mount the offline password guessing attack in order to know the password PW_i of U_i using the following steps as in [12]:

Step 1. Pick a guessed password PW_i^* from a relatively small dictionary as it is assumed in Amin-Biswas's scheme that PW_i is a low-entropy password.

Step 2. Compute $PWD_i^* = h(ID_i||PW_i^*)$.

Step 3. Check if $PWD_i^* = PWD_i$. If there is a match, the insider attacker is successful in finding the correct password PW_i of the user U_i and terminates the procedure. Otherwise, the insider attacker discards this guessed password and guesses another password, and continues from Step 1.

Hence, it is clear that Amin-Biswas's scheme is completely insecure against privileged-insider attack, where an insider user of the MRS being an attacker knows all the user's credentials ID_i, PW_i and B_i. In practice, it is expected that a user U_i keeps the same B_i and PW_i for different application due to easy-to-remember policy. In fact, PW_i and B_i must not be revealed even to the MRS though the MRS is considered as a trustworthy node in the network [41].

Fails to Protect Strong Replay Attack

Suppose an adversary $\mathcal{A}$ intercepts the login message $\langle ID_{MS_j}, ID_k, F_i, D_i, G_i, L_i \rangle$, which is sent to a medical server MS_j by a user U_i, during the login phase of Amin-Biswas's scheme. After that $\mathcal{A}$ sends the same intercepted message $\langle ID'_{MS_j}, ID'_k, F'_i, D'_i, G'_i, L'_i \rangle = \langle ID_{MS_j}, ID_k, F_i, D_i, G_i, L_i \rangle$ to the MS_j. However, this message is not detected as a replay message by the MRS due to the following reason. After receiving the login message, MS_j computes $E_i^* = h(ID_{MS_j}||X_j||F_i)$, $ID_i^* = G_i \oplus E_i^*$, $R_c^* = L_i \oplus E_i^*$, $C_i^* = h(ID_i^*||X_j)$ and $D_i^* = h(C_i^*||R_c^*)$. When MS_j checks the condition $D_i^* = D_i$, this condition is always satisfied. As a result, MS_j believes the authenticity of U_i and proceeds further to send the message $\langle ID_k, O_j, S_j, Q_j, RAN_j, F_i \rangle$ to a physician server PS_k. This happens, because there is no mechanism used in Amin-Biswas's scheme that the previous random nonce R_c is old or not. Thus, Amin-Biswas's scheme fails to protect strong replay attack.

Fails to Protect Strong Man-in-the-middle Attack

During the authentication and key agreement phase of Amin-Biswas's scheme, after receiving the login message $\langle ID_{MS_j}, ID_k, F_i, D_i, G_i, L_i\rangle$ from a user U_i, MS_j generates a random nonce R_{ms}, and computes $N_j = h(ID_k||X_k||F_i)$, $O_j = ID_i \oplus N_j$, $S_j = h(ID_i||X_k) \oplus R_{ms}$, $RAN_j = R_c^* \oplus R_{ms}$, $Q_j = h(ID_i||X_k||N_j||R_{ms})$. MS_j then sends the message $\langle ID_k, O_j, S_j, Q_j, RAN_j, F_i\rangle$ to the accessed physician server PS_k via a public channel. Suppose an adversary $\mathcal{A}$ intercepts the message $\langle ID_k, O_j, S_j, Q_j, RAN_j, F_i\rangle$, and generates a fake random nonce R_a and computes $RAN'_j = RAN_j \oplus R_a$, which is $R_c^* \oplus R_{ms} \oplus R_a$. $\mathcal{A}$ then sends the modified message $\langle ID_k, O_j, S_j, Q_j, RAN'_j, F_i\rangle$ to PS_k via a public channel. After receiving this modified message, PS_k proceeds to compute $N'_j = h(ID_k||X_k||F_i)$, $ID'_i = O_j \oplus N'_j$, $R'_{ms} = h(ID'_i||X_k) \oplus S_j$, $R'_c = RAN_j \oplus R'_{ms} \neq R_c$, $Q'_j = h(ID'_i||X_k||N'_j||R'_{ms})$. Then, the condition $Q'_j = Q_j$ is satisfied as this condition does not involve R_c. Thus, PS_k believes the authenticity of MS_j and U_i, and proceeds to generate a random nonce R_k and computes a session key $SK = h(ID'_i||ID_k||R'_c||R_k)$ shared with the user U_i, which is not equal to $h(ID_i||ID_k||R_c||R_k)$, $T_k = h(h(ID'_i||X_k)||SK)$, $RAN_k = R'_c \oplus R_k \neq R_c \oplus R_k$, $V_k = h(ID'_i||X_k) \oplus R_k$, and sends the message $\langle T_k, RAN_k, V_k\rangle$ to U_i via a public channel. At the user U_i's side, this message is then rejected. Thus, it is clear that Amin-Biswas's scheme gives the adversary $\mathcal{A}$ a fair chance to modify RAN_j to RAN'_j so that the condition $Q'_j = Q_j$ does not fail. Hence, Amin-Biswas's scheme also fails to protect strong man-in-the-middle attack.

Design Flaw in User Registration Phase

In Amin-Biswas's scheme, after the end of the user registration phase, the smart card does not store $F_i = H(B_i)$ for the purpose of biometric verification in the login phase. Also, the smart card contains the table having the tuples $\langle ID_{MS_j}, A_j, P_j\rangle$, $1 < j \leq m$. Thus, the smart card does not store the tuple $\langle ID_{MS_1}, A_1, P_1\rangle$ corresponding to the first medical server MS_1. This means that a user U_i can not access the medical services of the physician servers through the medical server MS_1.

Design Flaw in Login Phase

In Amin-Biswas's scheme, the biometric verification always fails as the condition $F_i^* = F_i$ will never meet, and $F_i = H(B_i)$ is not stored in the smart card's memory during the user registration phase as specified in Remark 1. Further, the smart card continues to compute $E_i = P_j \oplus h(REG_i||F_i)$, $G_i = ID_i \oplus E_i$ and $L_i = E_i \oplus R_c$, which contain the value of F_i. Therefore, without the actual value of F_i computation of E_i, G_i and L_i are incorrect and as a result, the login request message $\langle ID_{MS_j}, ID_k, F_i, D_i, G_i, L_i\rangle$ is totally incorrect. This is a very serious design flaw found in Amin-Biswas's scheme. In addition, during the password verification, the smart card computes $REG_i^* = h(ID_i||PW_i)$ and then checks the condition $REG_i^* = REG_i$. Note that $REG_i = h(ID_i||PWD_i) = h(ID_i||h(ID_i||PW_i))$. As a result, the password verification will also always fail in Amin-Biswas's scheme. It is thus clear that Amin-Biswas's scheme fails to verify both password as well as biometric verification during the login phase.

Design Flaw in Password Change Phase

In Amin-Biswas's scheme, the password change phase has the inherent flaw as found in the login phase (discussed in Section "Design Flaw in Login Phase"). In this phase, the biometric verification is not possible because the smart card does not store $F_i = H(B_i)$. Further, the password verification will also always fail in Amin-Biswas's scheme. In addition, P_j must be replaced by P_j^{new}, but not by B_j^{new} in the smart card's memory. As a consequence, the subsequent password change phases will be incorrect and subsequent authentication for the same user U_i by PS_k will always fail.

In the following, we show the offline password guessing attack in this phase. We assume that the smart card of the user U_i is lost or stolen and it is with an insider attacker (user) of the MRS. Note that the updated smart card contains the information $\{\{(ID_{MS_j}, A_j^{new}, B_j^{new})|1 < j \leq m\}, REG_i^{new}, h(\cdot), H(\cdot)\}$, where $REG_i^{new} = h(ID_i||PWD_i^{new}) = h(ID_i||h(ID_i||PW_i^{new}))$. Further, we assume that the insider attacker of the MRS has the registration request message $\langle ID_i, PWD_i, B_i\rangle$ earlier. Thus, the insider attacker knows the identity ID_i of the user U_i. The offline password guessing attack in Amin-Biswas's scheme works as follows. Note that this attack is similar to that presented in [12].

Step 1. Extract the information stored in the lost/stolen smart card of the user U_i using the power analysis attack [38] as described in our threat model. Thus, the insider attacker knows $REG_i^{new} = h(ID_i||PWD_i^{new}) = h(ID_i||h(ID_i||PW_i^{new}))$.

Step 2. Pick a guessed password PW_i^*.

Step 3. Compute $PWD_i^* = h(ID_i||PW_i^*)$ and $REG_i^* = h(ID_i||PWD_i^*) = h(ID_i||h(ID_i||PW_i^*))$.

Step 4. Check if $REG_i^* = REG_i^{new}$. If there is a match, the insider attacker is successful in finding the correct new updated password PW_i^{new} of the user U_i and terminates the procedure. Otherwise, the insider attacker discards this guessed password and guesses another new password, and continues from Step 2.

Since in Amin-Biswas's scheme, the password is assumed to be low-entropy, it is very clear from the above attack that the insider user being an attacker can successfully obtain the new changed password even if the old password of the user U_i is changed through the password change phase.

Lacks of Supporting Biometric Update Phase

In a biometrics-based authentication scheme, one of the most desirable properties is that a legal user must be allowed to change his/her old password as well as biometrics [12, 41]. However, it is noted that Amin-Biswas's scheme fails to support biometric update phase.

Flaws in Formal Security Analysis

Amin and Biswas used the *Reveal* oracle for their formal security analysis. It is an oracle, which unconditionally outputs the input string m from the corresponding hash value $y = h(m)$. In the first algorithm, they have used the *Reveal* oracle on the input $REG_i = h(ID_i||PWD_i) = h(ID_i||h(ID_i||PW_i))$ in order to retrieve ID_i, PW_i and B_i as $(ID_i'||PW_i'||B_i') \leftarrow Reveal(REG_i)$. However, since REG_i does not involve B_i, the *Reveal* oracle never returns the biometrics B_i'. Hence, B_i' can not be used in the rest of the algorithm. As a result, the formal security analysis is incorrect in Amin-Biswas's scheme.

The Proposed Scheme

In this section, we describe the various phases of our scheme, which are (i) medical server registration phase, (ii) user registration phase, (iii) login phase, (iv) authentication and session key agreement phase, (v) password and biometric update phase, and (vi) dynamic medical server addition phase. Our scheme is an improvement over Amin-Biswas's scheme, which withstands all the security pitfalls and design flaws found in their scheme, which are already discussed in the previous section. We use the notations provided in Table 1 for describing our scheme. We also use the same architecture provided in Fig. 1 for describing our improved scheme.

We apply the fuzzy extractor algorithms in order to strengthen the biometric verification procedure in our scheme. Further, we make use of both the time-stamp and random nonce to protect strongly the replay and man-in-the-middle attacks. For this reason, we assume that all the network entities, such as the MRS, medical servers MS_j $(1 \leq j \leq m + m')$, physician servers PS_k $(1 \leq k \leq p)$ and user devices (smart cards) are synchronized with their clocks. The various phases of our scheme are discussed in the following subsections.

Medical Server Registration Phase

Suppose m number of medical servers MS_j, $(1 \leq j \leq m)$ are to be deployed initially in the network. We further assume that m' number of additional medical servers MS_j, $(m + 1 \leq j \leq m + m')$ may be added later in the network, where $m' << m$. For example, initially $m = 100$ medical servers may be deployed and later we may add $m' = 10$ additional medical servers after initial deployment in the network, if required, based on the demand of the medical services when more users want to access the services.

For this purpose, a medical server MS_j, $(1 \leq j \leq m)$, which wants to provide the medical services to the remote users (patients), needs to select a unique identity ID_{MS_j} and send it to the MRS. After receiving ID_{MS_j}, the MRS computes the secret key $X_j = h(ID_{MS_j}||X_c)$, where X_c is the 1024-bit secret key of the MRS for security reasons, and sends it back to MS_j via a secure channel. Thus, each MS_j keeps (ID_{MS_j}, X_j). For m' additional medical servers MS_l, $(m + 1 \leq l \leq m + m')$, the MRS itself chooses a unique identity ID_{MS_l} and also compute the secret key $X_l = h(ID_{MS_l}||X_c)$. Note that these computed (ID_{MS_l}, X_l) are kept to the MRS and will be used later during the user registration phase and dynamic medical server addition phase.

User Registration Phase

In this phase, a legal user U_i needs to register with the MRS for accessing the medical services from a particular physician server PS_k under a medical server MS_j in the network. This phase has the following steps:

Step R1: As in [13], U_i first inputs his/her desired identity ID_i, password PW_i, and then imprints the personal biometrics B_i at the sensor of a specific device. U_i generates a 1024-bit random number K, which is kept secret to U_i only. U_i then applies the fuzzy extractor generation function $Gen(\cdot)$ on the input B_i in order to produce the biometric data key σ_i and the public parameter τ_i as $Gen(B_i) = (\sigma_i, \tau_i)$. Note that σ_i is kept secret to U_i only.

Step R2: U_i computes the pseudo-random password RPW_i as $RPW_i = h(ID_i||K||PW_i)$ and sends the registration request $\langle ID_i, RPW_i \rangle$ to the MRS via a secure channel.

Step R3: After receiving the registration request from U_i, the MRS continues to compute $A_j = h(ID_i||X_j) \oplus RPW_i$ and $P_j = h(ID_{MS_j}||X_j) \oplus RPW_i$, for $1 \leq j \leq m + m'$. Then the MRS stores the information $\{\{\langle ID_{MS_j}, A_j, P_j \rangle | 1 \leq j \leq m + m'\}, h(\cdot), Gen(\cdot), Rep(\cdot), t\}$ in a smart card, say SC_i and sends it to the user U_i via a secure channel, where t is the error tolerance threshold used in fuzzy extractor.

Step R4: After receiving the smart card SC_i from the MRS, the user U_i computes $e_i = h(ID_i||\sigma_i) \oplus K$ and $f_i = h(ID_i||RPW_i||\sigma_i)$. U_i then stores e_i and f_i in the smart card SC_i. Finally, note that the smart card SC_i contains the information $\{\langle ID_{MS_j}, A_j, P_j\rangle | 1 \leq j \leq m + m'\}$, e_i, f_i, $h(\cdot)$, $Gen(\cdot)$, $Rep(\cdot)$, τ_i, and t.

The summary of the user registration phase of our scheme is shown in Table 4.

Login phase

In this phase, a legal user U_i can access any medical server MS_j for the medical services from a physician server PS_k under that medical server MS_j at anytime from anywhere through his/her issued smart card SC_i. This phase contains the following steps:

Step L1: U_i first inserts his/her smart card SC_i into a smart card reader of a specific terminal, and then inputs his/her identity ID_i, password PW_i, and also imprints the personal biometrics B_i at the sensor.

Step L2: SC_i then computes

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

SC_i further checks the verification condition $f_i^* = f_i$. If it holds, it ensures that the user U_i passes successfully both password and biometric verification. Otherwise, this phase is terminated immediately.

Step L3: SC_i further proceeds to generate a random nonce R_c and the current time-stamp TS_c. Then SC_i computes

$$\begin{aligned} M_1 &= A_j \oplus RPW_i^* \\ &= h(ID_i||X_j) \oplus RPW_i \oplus RPW_i^* \\ &= h(ID_i||X_j), \\ M_2 &= P_j \oplus RPW_i^* \\ &= h(ID_{MS_j}||X_j), \\ M_3 &= ID_i \oplus M_2, \\ M_4 &= ID_i \oplus M_1 \oplus R_c, \\ M_5 &= h(M_1||M_3||M_4||R_c||TS_c). \end{aligned}$$

SC_i sends the login request message $\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c\rangle$ to the medical server MS_j via a public channel, where ID_k is the identity of the physician server PS_k from where U_i wants to access the medical service.

The summary of the login phase of our scheme is shown in Table 5.

Authentication and Session key Agreement Phase

In this phase, a legal user U_i authenticates an accessed physician server PS_k and PS_k also authenticates U_i for mutual authentication purpose before they can establish a symmetric common session key SK_{U_i,PS_k} between them for their future secure communication. This phase involves the following steps:

Step A1: $\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c\rangle$ from U_i, MS_j verifies the validity of the received time-stamp TS_c in the message. Let the login request message be received by MS_j at time TS_c^*. MS_j then checks the condition $|TS_c^* - TS_c| \leq \Delta T$, where ΔT denotes the maximum transmission delay or preset acceptable delay threshold or expected time interval for transmission delay or

Table 4 User registration phase of our scheme

U_i	MRS
Input ID_i and PW_i, and imprint B_i.	
Compute $(\sigma_i, \tau_i) = Gen(B_i)$	
Generate a 1024-bit random number K.	
Compute $RPW_i = h(ID_i\|\|K\|\|PW_i)$.	
$\langle ID_i, RPW_i\rangle \xrightarrow{}$ (via a secure channel)	
	Compute $A_j = h(ID_i\|\|X_j) \oplus RPW_i$, $P_j = h(ID_{MS_j}\|\|X_j) \oplus RPW_i$, for $1 \leq j \leq m + m'$.
	$\langle$Smart Card$(\{\langle ID_{MS_j}, A_j, P_j\rangle \| 1 \leq j \leq m + m'\}, h(\cdot), Gen(\cdot), Rep(\cdot), t)\rangle \xleftarrow{}$ (via a secure channel)
Compute $e_i = h(ID_i\|\|\sigma_i) \oplus K$, $f_i = h(ID_i\|\|RPW_i\|\|\sigma_i)$. Store $\{e_i, f_i, \tau_i\}$ in smart card, SC_i.	

Table 5 Login phase of our scheme

U_i/SC_i	MS_j
Insert the smart card SC_i and input ID_i, PW_i, and B_i.	
Compute $\sigma_i^* = Rep(B_i, \tau_i)$, $K^* = h(ID_i \| \sigma_i^*) \oplus e_i$, $RPW_i^* = h(ID_i \| K^* \| PW_i)$, $f_i^* = h(ID_i \| RPW_i^* \| \sigma_i^*)$. Check if $f_i^* = f_i$? If so, generate R_c and TS_c. Compute $M_1 = A_j \oplus RPW_i^*$, $M_2 = P_j \oplus RPW_i^*$, $M_3 = ID_i \oplus M_2$, $M_4 = ID_i \oplus M_1 \oplus R_c$, $M_5 = h(M_1 \| M_3 \| M_4 \| R_c \| TS_c)$.	
$\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c \rangle$ (via a public channel)	

expected network delay time. If this condition fails, the login request message is rejected and also the session is terminated immediately. Otherwise, MS_j executes the next step.

Step A2: MS_j continues to compute $M_6 = h(ID_{MS_j} \| X_j)$ using its own identity ID_{MS_j} and the secret key X_j, where $X_j = h(ID_{MS_j} \| X_c)$ and X_c is the secret key of the MRS. MS_j then computes

$$\begin{aligned} M_7 &= M_3 \oplus M_6 \\ &= ID_i, \\ M_8 &= h(M_7 \| X_j) \\ &= h(ID_i \| X_j), \\ M_9 &= M_4 \oplus M_7 \oplus M_8 \\ &= R_c, \\ M_{10} &= h(M_8 \| M_3 \| M_4 \| M_9 \| TS_c) \\ &= h(h(ID_i \| X_j) \| M_3 \| M_4 \| R_c \| TS_c). \end{aligned}$$

MS_j further checks the condition $M_{10} = M_5$. If it holds, MS_j believes the authenticity of the user U_i. Otherwise, MS_j terminates the session immediately.

In order to protect strongly the replay and man-in-the-middle attacks in our scheme, we adopt the following strategy as suggested in [11, 33]. If the condition $M_{10} = M_5$ holds, MS_j stores the pair $(M_7, M_9) = (ID_i, R_c)$ in its database. Later, when MS_j receives the next login request message, say $\langle ID_{MS_j}, ID_k, M_3', M_4', M_5', TS_c' \rangle$, MS_j first checks the validity of the time-stamp TS_c'. If it is valid, MS_j computes $M_6' = h(ID_{MS_j} \| X_j)$, $M_7' = M_3' \oplus M_6'$, $M_8' = h(M_7' \| X_j)$, $M_9' = M_4' \oplus M_7' \oplus M_8'$. After that MS_j compares M_9' with the stored $M_9 = R_c$ corresponding to the user U_i's identity $M_7 = ID_i$ in its database. If there is a match, MS_j ensures that the received login request message $\langle ID_{MS_j}, ID_k, M_3', M_4', M_5', TS_c' \rangle$ is a replay message and discards this message. Otherwise, MS_j replaces M_9 with M_9' in its database and treats this message as a fresh message.

Step A3: MS_j generates a random nonce R_{ms} and the current time-stamp TS_{ms}. MS_j computes $M_{11} = h(ID_{MS_j} \| ID_k \| X_k)$, where X_k is the secret key shared between MS_j and PS_k. MS_j further computes

$$\begin{aligned} M_{12} &= ID_i \oplus M_{11}, \\ M_{13} &= h(ID_i \| X_k) \oplus R_{ms}, \\ M_{14} &= ID_i \oplus M_9 \oplus R_{ms} \\ &= ID_i \oplus R_c \oplus R_{ms}, \\ M_{15} &= h(ID_i \| M_{11} \| M_{12} \| M_{13} \| M_{14} \\ &\quad \| M_9 \| R_{ms} \| TS_{ms}). \end{aligned}$$

MS_j then sends the authentication request message $\langle ID_{MS_j}, ID_k, M_{12}, M_{13}, M_{14}, M_{15}, TS_{ms} \rangle$ to the physician server PS_k via a public channel.

Step A4: After receiving the message in Step A3, PS_k checks the validity of the received time-stamp TS_{ms} in the message by the condition $|TS_{ms}^* - TS_{ms}| \leq \triangle T$, where TS_{ms}^* is the time when the message is received by PS_k. If it is valid, PS_k further continues to compute

$$\begin{aligned} M_{16} &= h(ID_{MS_j} \| ID_k \| X_k), \\ M_{17} &= M_{12} \oplus M_{16} \\ &= ID_i, \\ M_{18} &= M_{13} \oplus h(M_{17} \| X_k) \\ &= R_{ms}, \\ M_{19} &= M_{14} \oplus M_{17} \oplus M_{18} \\ &= R_c, \\ M_{20} &= h(M_{17} \| M_{16} \| M_{12} \| M_{13} \| \\ &\quad M_{14} \| M_{19} \| M_{18} \| TS_{ms}) \\ &= h(ID_i \| h(ID_{MS_j} \| ID_k \| X_k) \| M_{12} \| \\ &\quad M_{13} \| M_{14} \| R_c \| R_{ms} \| TS_{ms}). \end{aligned}$$

PS_k then checks the condition $M_{20} = M_{15}$. If it does not hold, the session is terminated by PS_k. Otherwise, PS_k believes the authenticity of both MS_j as well as U_i.

Step A5: PS_k generates a random nonce R_k and the current time-stamp TS_k. PS_k also computes

$$\begin{aligned} M_{21} &= h(M_{17}||X_k) \\ &= h(ID_i||X_k), \\ M_{22} &= M_{17} \oplus M_{19} \oplus R_k \\ &= ID_i \oplus R_c \oplus R_k, \\ M_{23} &= M_{21} \oplus R_k \\ &= h(ID_i||X_k) \oplus R_k, \\ SK_{U_i,PS_k} &= h(M_{17}||ID_k||M_{19}||R_k||M_{21}||TS_k) \\ &= h(ID_i||ID_k||R_c||R_k||h(ID_i||X_k)||TS_k), \\ M_{24} &= h(SK_{U_i,PS_k}||M_{22}||M_{23}||M_{19}||R_k||TS_k). \end{aligned}$$

PS_k finally sends the authentication reply message $\langle ID_k, M_{22}, M_{23}, M_{24}, TS_k\rangle$ to the user U_i via a public channel.

Step A6: After receiving the message in Step A5, the smart card SC_i of the user U_i checks the validity of the time-stamp TS_k in the received message by the condition $|TS_k^* - TS_k| \leq \triangle T$, where TS_k^* is the time when the message is received by U_i. If it holds, U_i computes

$$\begin{aligned} M_{25} &= M_{22} \oplus (ID_i \oplus R_c) \\ &= R_k, \\ M_{26} &= M_{23} \oplus M_{25} \\ &= h(ID_i||X_k), \\ SK^*_{U_i,PS_k} &= h(ID_i||ID_k||R_c||M_{25}||M_{26}||TS_k), \\ M_{27} &= h(SK^*_{U_i,PS_k}||M_{22}||M_{23}||R_c||M_{25}||TS_k). \end{aligned}$$

SC_i then checks if $M_{27} = M_{24}$. If it matches, U_i authenticates PS_k, and both U_i and PS_k treat $SK^*_{U_i,PS_k} = SK_{U_i,PS_k}$ as the session key shared between them.

The summary of the authentication and session key agreement phase of our scheme is shown in Table 6.

Password and Biometric Update Phase

This phase is executed by a legal user U_i, if he/she wants to change his/her old password and biometrics for some security reasons. Note that this phase is not executed frequently by U_i. This phase has the following steps:

Step PB1: U_i first inserts his/her smart card SC_i into a smart card reader of a specific terminal, and then inputs his/her identity ID_i, old password PW_i^{old}, and also imprints the old personal biometrics B_i^{old} at the sensor. The smart card SC_i of U_i computes

$$\begin{aligned} \sigma_i^{old} &= Rep(B_i^{old}, \tau_i), \\ K' &= h(ID_i||\sigma_i^{old}) \oplus e_i, \\ RPW_i^{old} &= h(ID_i||K'||PW_i^{old}), \\ f_i^{old} &= h(ID_i||RPW_i^{old}||\sigma_i^{old}), \end{aligned}$$

and checks the verification condition $f_i^{old} = f_i$. If it holds, it ensures that U_i passes successfully both old password and biometric verification. Otherwise, this phase is terminated immediately.

Step PB2: SC_i asks the user U_i to input new password and biometrics. Let PW_i^{new} and B_i^{new} be the new password and personal biometrics entered by U_i. SC_i then applies the fuzzy extractor generation function $Gen(\cdot)$ on the input B_i^{new} in order to produce the biometric data key σ_i^{new} and the public parameter τ_i^{new} as $Gen(B_i^{new}) = (\sigma_i^{new}, \tau_i^{new})$, where σ_i^{new} is kept secret to U_i only.

Step PB3: SC_i computes $RPW_i^{new} = h(ID_i||K'||PW_i)$, $e_i^{new} = h(ID_i||\sigma_i^{new}) \oplus K'$ and $f_i^{new} = h(ID_i||RPW_i^{new}||\sigma_i^{new})$. SC_i further computes $\{(A_j^{new}, P_j^{new})|1 \leq j \leq m + m'\}$, where $A_j^{new} = A_j \oplus RPW_i^{old} \oplus RWP_i^{new} = h(ID_i||X_j) \oplus RPW_i^{new}$ and $P_j^{new} = P_j \oplus RPW_i^{old} \oplus RWP_i^{new} = h(ID_{MS_j}||X_j) \oplus RPW_i^{new}$.

Step PB4: Finally, SC_i replaces $\{\langle A_j, P_j\rangle|1 \leq j \leq m + m'\}$, e_i, f_i,and τ_i with $\{\langle A_j^{new}, P_j^{new}\rangle|1 \leq j \leq m + m'\}$, e_i^{new}, f_i^{new},and τ_i^{new}, respectively, in its memory. As a result, the updated smart card SC_i now contains the information $\{\langle ID_{MS_j}, A_j^{new}, P_j^{new}\rangle|1 \leq j \leq m + m'\}$, e_i^{new}, f_i^{new}, $h(\cdot)$, $Gen(\cdot)$, $Rep(\cdot)$, τ_i^{new},and t.

It is thus clear that the new password and biometric are correctly updated in the user U_i's smart card SC_i only after validating the old password and biometric. This phase is executed locally without the involvement of the MRS further by the user U_i.

Dynamic Medical Server Addition Phase

It is assumed in the medical server registration phase that m medical servers MS_j $(1 \leq j \leq m)$ are deployed in the network. If later some more m' medical servers MS_j $(m + 1 \leq j \leq m + m')$ are deployed, it is one of the desirable requirement that the proposed scheme should support dynamic medical server addition after initial deployment. Our scheme has the ability to support this important feature and it is explained below.

Note that during our medical server registration phase, for m' additional medical servers MS_l, $(m + 1 \leq l \leq m+m')$, the MRS already selected a unique identity ID_{MS_l} and computed the secret key $X_l = h(ID_{MS_l}||X_c)$. Assume that a new medical server MS_l be deployed in the existing network. Then, MS_j needs to register with the MRS. In this case, MS_l needs to simply send a request to the MRS and the information $\langle ID_{MS_l}, X_l\rangle$ are sent back to MS_l securely by the MRS. Thus, MS_l does not choose any identity in this phase. MS_l then stores the information $\{ID_{MS_l}, X_l\}$. Further, note that during the user registration phase, a legal registered user U_i's smart card SC_i already contains the

Table 6 Authentication and session key agreement phase of our scheme

U_i/SC_i	MS_j	PS_k
	Check the validity of TS_c in the received login message. If valid, compute $M_7 = M_3 \oplus M_6$, $M_8 = h(M_7 \| X_j)$, $M_9 = M_4 \oplus M_7 \oplus M_8$, $M_{10} = h(M_8 \| M_3 \| M_4 \| M_9 \| TS_c)$. Check if $M_{10} = M_5$. If so, MS_j believes the authenticity of the user U_i. Generate R_{ms} and TS_{ms}. Compute $M_{11} = h(ID_{MS_j} \| ID_k \| X_k)$, $M_{12} = ID_i \oplus M_{11}$, $M_{13} = h(ID_i \| X_k) \oplus R_{ms}$, $M_{14} = ID_i \oplus M_9 \oplus R_{ms}$, $M_{15} = h(ID_i \| M_{11} \| M_{12} \| M_{13} \| M_{14} \| M_9 \| R_{ms} \| TS_{ms})$. $\langle ID_{MS_j}, ID_k, M_{12}, M_{13}, M_{14}, M_{15}, TS_{ms} \rangle$ $\longrightarrow$ (via a public channel)	Check the validity of TS_{ms} in the received authentication request message. If valid, compute $M_{16} = h(ID_{MS_j} \| ID_k \| X_k)$, $M_{17} = M_{12} \oplus M_{16}$, $M_{18} = M_{13} \oplus h(M_{17} \| X_k)$, $M_{19} = M_{14} \oplus M_{17} \oplus M_{18}$, $M_{20} = h(M_{17} \| M_{16} \| M_{12} \| M_{13} \| M_{14} \| M_{19} \| M_{18} \| TS_{ms})$. Check if $M_{20} = M_{15}$. If so, PS_k believes the authenticity of both MS_j as well as U_i. Generate R_k and TS_k. Compute $M_{21} = h(M_{17} \| X_k)$, $M_{22} = M_{17} \oplus M_{19} \oplus R_k$, $M_{23} = M_{21} \oplus R_k$, $SK_{U_i,PS_k} = h(M_{17} \| ID_k \| M_{19} \| R_k \| M_{21} \| TS_k)$, $M_{24} = h(SK_{U_i,PS_k} \| M_{22} \| M_{23} \| M_{19} \| R_k \| TS_k)$.
	$\langle ID_k, M_{22}, M_{23}, M_{24}, TS_k \rangle$ $\longleftarrow$	(via a public channel to U_i)
Check the validity of TS_k in the received authentication reply message from PS_k. If valid, compute $M_{25} = M_{22} \oplus (ID_i \oplus R_c)$, $M_{26} = M_{23} \oplus M_{25}$, $SK^*_{U_i,PS_k} = h(ID_i \| ID_k \| R_c \| M_{25} \| M_{26} \| TS_k)$, $M_{27} = h(SK^*_{U_i,PS_k} \| M_{22} \| M_{23} \| R_c \| M_{25} \| TS_k)$. Check if $M_{27} = M_{24}$. If valid, U_i authenticates PS_k. Store $SK^*_{U_i,PS_k}$ as session key shared with PS_k.		Store SK_{U_i,PS_k} as session key shared with U_i.

information regarding the new medical server MS_l, since SC_i has the information $\{\langle ID_{MS_j}, A_j, P_j \rangle | 1 \leq j \leq m + m'\}$, e_i, f_i, $h(\cdot)$, $Gen(\cdot)$, $Rep(\cdot)$, τ_i, and t. Thus, there is no need to update the user's smart card in this phase. Finally, the MRS needs to inform the user U_i regarding the addition of new medical server MS_l so that U_i can access services of some physician servers through this medical server MS_l.

Security Analysis of the Proposed Scheme

In this section, we show that our scheme is secure against various known attacks.

Formal Security Analysis using BAN Logic

In this section, we provide the authentication proof using the widely-accepted Burrows-Abadi-Needham (BAN) logic [6]. In this proof, we show that both a legal user U_i and a physician server PS_k belonging under a medical server MS_j mutually authenticate among each other.

The notations used in the BAN logic are given below:

$P \mid\equiv X$: Principal P believes a statement X or P is entitled to believe X.

$\#(X)$: Formula X is considered as fresh.

$P \mid\Rightarrow X$: Principal P has jurisdiction over statement X.

$P \lhd X$: Principal P sees the statement X.

$P \mid\sim X$: Principal P once said the statement X.
(X, Y) : Formula X or Y is a part of formula (X, Y).
$\{X\}_K$: Formula X is encrypted under the key K.
$\langle X \rangle_Y$: Formula X is combined with the formula Y.
$P \overset{K}{\longleftrightarrow} Q$: P and Q may use the shared key K to communicate. The key K is good, in that it will never be discovered by any principal except P and Q.
$P \overset{X}{\rightleftharpoons} Q$: Formula X is secret known only to P and Q, and possibly to principals trusted by them.

Rules: We have the following four rules used in the BAN logic:

- $Rule(1)$. Message-meaning rule: $\frac{P|\equiv P \overset{K}{\longleftrightarrow} Q, P \lhd \{X\}_K}{P|\equiv Q|\sim X}$ and $\frac{P|\equiv P \overset{Y}{\rightleftharpoons} Q, P \lhd \langle X \rangle_Y}{P|\equiv Q|\sim X}$
- $Rule(2)$. Nonce-verification rule: $\frac{P|\equiv \#(X), P|\equiv Q|\sim X}{P|\equiv Q|\equiv X}$
- $Rule(3)$. Jurisdiction rule: $\frac{P|\equiv Q|\Rightarrow X, P|\equiv Q|\equiv X}{P|\equiv X}$
- $Rule(4)$. Freshness-conjuncatenation rule: $\frac{P|\equiv \#(X)}{P|\equiv \#(X,Y)}$

Goals: According to the analytic procedures of the BAN logic, the proposed protocol must satisfy the following test goals in order to ensure the system is secure. Assume that SK is shared session key SK_{U_i, PS_k} between U_i and PS_k for the sake of simplicity of the proof.

- $G_1 : U_i \mid\equiv U_i \overset{SK}{\rightleftharpoons} PS_k$.
- $G_2 : PS_k \mid\equiv U_i \overset{SK}{\rightleftharpoons} PS_k$.

Idealized form: The arrangement of the proposed protocol to its idealized forms are as follows:

- From the login request message (Step L3), we have $U_i \rightarrow MS_j : \langle ID_i \oplus h(ID_{MS_j}, X_j), ID_i \oplus M_1 \oplus R_c, R_c, TS_c \rangle_{U_i \overset{M_1}{\longleftrightarrow} MS_j}$.
- From the authentication request message (Step A3), we have $MS_j \rightarrow PS_k : \langle ID_i, h(ID_{MS_j}, ID_k, X_k), ID_i \oplus M_{11}, h(ID_i, X_k) \oplus R_{ms}, ID_i \oplus R_c \oplus R_{ms}, U_i \overset{R_c}{\rightleftharpoons} PS_k, R_{ms}, TS_{ms} \rangle_{MS_j \overset{X_k}{\longleftrightarrow} PS_k}$.
- From the authentication reply message (Step A5), we have $PS_k \rightarrow U_i : \langle U_i \overset{SK}{\longleftrightarrow} PS_k, ID_i \oplus R_c \oplus R_k, h(ID_i \oplus X_k) \oplus R_k, U_i \overset{R_k}{\longleftrightarrow} PS_k, TS_k \rangle_{U_i \overset{R_c}{\longleftrightarrow} PS_k}$.

Hypotheses: The following assumptions about the initial state are made to analyze the proposed protocol:

- $H_1 : U_i \mid\equiv \#(TS_k), MS_j \mid\equiv \#(TS_c), PS_k \mid\equiv \#(TS_{ms})$.
- $H_2 : U_i \mid\equiv U_i \overset{M_1}{\longleftrightarrow} MS_j$.
- $H_3 : MS_j \mid\equiv U_i \overset{M_1}{\longleftrightarrow} MS_j$.
- $H_4 : MS_j \mid\equiv MS_j \overset{X_k}{\longleftrightarrow} PS_k$.
- $H_5 : PS_k \mid\equiv MS_j \overset{X_k}{\longleftrightarrow} PS_k$.
- $H_6 : PS_k \mid\equiv MS_j \mid\Rightarrow U_i \mid\sim X$.
- $H_7 : U_i \mid\equiv PS_k \mid\Rightarrow U_i \overset{R_k}{\longleftrightarrow} PS_k$.
- $H_8 : MS_j \mid\equiv U_i \mid\Rightarrow U_i \overset{R_c}{\longleftrightarrow} PS_k$.
- $H_9 : PS_k \mid\equiv U_i \mid\Rightarrow U_i \overset{R_c}{\longleftrightarrow} PS_k$.

The idealized form of the proposed protocol is analyzed based on the BAN logic rules and the assumptions. The main proofs are stated as follows:

- From the login request message, we have $S_1 : MS_j \lhd \langle ID_i \oplus h(ID_{MS_j}, X_j), ID_i \oplus M_1 \oplus R_c, U_i \overset{R_c}{\rightleftharpoons} PS_k, TS_c \rangle_{U_i \overset{M_1}{\longleftrightarrow} MS_j}$.
- From S_1, H_1, H_3, $Rule(1)$, $Rule(2)$ and $Rule(4)$, we get $S_2 : MS_j \mid\equiv U_i \mid\sim R_c$. MS_j shares R_c with PS_k as it is shared by the legal user U_i in the authentication request message.
- From the authentication request message, we have $S_3 : PS_k \lhd \langle ID_i, h(ID_{MS_j}, ID_k, X_k), ID_i \oplus M_{11}, h(ID_i, X_k) \oplus R_{ms}, ID_i \oplus R_c \oplus R_{ms}, U_i \overset{R_c}{\rightleftharpoons} PS_k, R_{ms}, TS_{ms} \rangle_{MS_j \overset{X_k}{\longleftrightarrow} PS_k}$.
- From S_3, H_1, H_5, $Rule(1)$, $Rule(2)$ and $Rule(4)$, we get $S_4 : PS_k \mid\equiv MS_j \mid\equiv (U_i \overset{R_c}{\rightleftharpoons} PS_k))$.
- From S_4 and $Rule(3)$, we get $S_5 : PS_k \mid\equiv U_i \overset{R_c}{\rightleftharpoons} PS_k$. Since the session key SK is computed using the formula $SK = SK_{U_i, PS_k} = h(M_{17}||ID_k||M_{19}||R_k||M_{21}||TS_k) = h(ID_i||ID_k||R_c||R_k||h(ID_i||X_k)||TS_k)$, where R_k is random number generated by PS_k, we get the goal G_1. This implies that $PS_k \mid\equiv U_i \overset{SK}{\rightleftharpoons} PS_k$, which is **Goal** G_2. From S_5, it is clear that the random secret is shared between U_i and PS_k.
- From the authentication reply message, we have $S_6 : U_i \lhd \langle U_i \overset{SK}{\rightleftharpoons} PS_k, ID_i \oplus R_c \oplus R_k, h(ID_i \oplus X_k) \oplus R_k, U_i \overset{R_k}{\rightleftharpoons} PS_k, TS_k \rangle_{U_i \overset{R_c}{\rightleftharpoons} PS_k}$.
- From S_5, S_6, H_1, H_3, H_7, $Rule(1)$, $Rule(2)$ and $Rule(4)$, and since the session key SK is computed

as the formula using R_c and R_k, that is, $SK_{U_i,PS_k} = h(ID_i||ID_k||R_c||R_k||h(ID_i||X_k)||TS_k)$, we have the **Goal** G_1 as $S_7 : U_i \mid\equiv U_i \overset{SK}{\rightleftharpoons} PS_k$.

This completes the proof.

Formal Security Analysis using Random Oracle Model

In this section, we perform the formal security analysis of our scheme under the random oracle model as in [7, 13, 41, 48] for the formal security analysis. We apply the method of contradiction proof [10] for our formal security analysis. Note that one can also prove the formal security in the standard model. However, in this paper, we perform the formal security analysis under the generic group model of cryptography.

In order to apply the method of contradiction proof [10], we assume that the following random oracle exists for an adversary, say $\mathcal{A}$:

– *Reveal* : This random oracle will unconditionally output the input string x from the corresponding hash value $y = h(x)$.

Theorem 1 *Under the assumption that a one-way hash function $h(\cdot)$ closely behaves like a random oracle, our scheme is secure against an adversary for deriving the identity ID_i of a legal user U_i and the secret key X_j of a medical server MS_j.*

Proof This proof is similar to that presented in [7, 13, 14, 16, 41, 48]. In this proof, we plan to construct an adversary $\mathcal{A}$ who will have the ability to derive the identity ID_i of a legal user U_i and the secret key X_j of a medical server MS_j. For this purpose, $\mathcal{A}$ can use the *Reveal* oracle and run the experiment $EXP1$ for our scheme given in Algorithm 1. We define the success probability for $EXP1$ as $Succ1 = |Pr[EXP1 = 1] - 1|$. The advantage function for this experiment becomes $Adv1(et_1, q_R) = \max_{\mathcal{A}}\{Succ1\}$, where the maximum is taken over all $\mathcal{A}$ with execution time et_1, and the number of queries q_R made to the *Reveal* oracle. Our scheme is said to be provably secure against $\mathcal{A}$ for deriving ID_i and X_j, if $Adv1\ (et_1, q_R) \leq \epsilon_1$, for any sufficiently small $\epsilon_1 > 0$. According to the experiment provided in Algorithm 1, if $\mathcal{A}$ has the ability to invert the one-way cryptographic hash function $h(\cdot)$, he/she can easily derive ID_i and X_j, and win the game. However, by Definition 1, it is a computationally infeasible problem to invert $h(\cdot)$, that is, $Adv_{\mathcal{A}}^{HASH}(t) \leq \epsilon$, for any sufficiently small $\epsilon > 0$. Since $Adv1\ (et_1, q_R)$ depends on the advantage $Adv_{\mathcal{A}}^{HASH}(t)$, we have $Adv1\ (et_1, q_R) \leq \epsilon_1$. Hence, our scheme is secure against an adversary for deriving the identity ID_i of a legal user U_i and the secret key X_j of a medical server MS_j. □

Algorithm 1 EXP1

1: Eavesdrop the login request message $\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c\rangle$ during the login phase.
2: Call *Reveal* oracle on input M_5 to retrieve the information M_1, M_3, M_4, R_c,and TS_c as $(M_1'||M_3'||M_4'||R_c'||TS_c') \leftarrow Reveal(M_5)$.
3: **if** $(TS_c' = TS_c)$ and $(M_3' = M_3)$ and $(M_4' = M_4)$
4: Compute $ID_i' = M_4 \oplus (M_1 \oplus R_c')$.
5: Call *Reveal* oracle on input M_1' to retrieve ID_i and X_j as $(ID_i''||X_j'') \leftarrow Reveal(M_1')$.
6: **if** $(ID_i'' = ID_i')$
7: Accept ID_i' and X_j' as the correct identity and secret key of the user U_i and the medical server MS_j, respectively.
8: **return** 1 (Success)
9: **else**
10: **return** 0 (Failure)
11: **end if**
12: **else**
13: **return** 0 (Failure)
14: **end if**

Theorem 2 *Under the assumption that a one-way hash function $h(\cdot)$ closely behaves like a random oracle, our scheme is secure against an adversary for deriving the secret key X_k of a physician server PS_k.*

Proof This proof is similar to that in Theorem 1. We construct an adversary $\mathcal{A}$ who will have the ability to derive successfully the secret key X_k of a physician server PS_k. For this purpose, $\mathcal{A}$ can use the *Reveal* oracle and run the experiment $EXP2$ provided in Algorithm 1.

We define the success probability for $EXP2$ as $Succ2 = |Pr[EXP2 = 1] - 1|$ and the advantage function as $Adv2(et_2, q_R) = \max_{\mathcal{A}}\{Succ2\}$, where the maximum is taken over all $\mathcal{A}$ with execution time et_2, and the number of queries q_R made to the *Reveal* oracle. Our scheme is said to be provably secure against $\mathcal{A}$ for deriving X_k, if we have $Adv2\ (et_2, q_R) \leq \epsilon_2$, for any sufficiently small $\epsilon_2 > 0$. According to this experiment, if $\mathcal{A}$ has the ability to invert the one-way cryptographic hash function $h(\cdot)$, he/she can easily derive X_k, and win the game. However, by Definition 1, it is again a computationally infeasible problem to invert $h(\cdot)$, that is, $Adv_{\mathcal{A}}^{HASH}(t) \leq \epsilon$, for any sufficiently small $\epsilon > 0$. Since $Adv2\ (et_2, q_R)$ depends on the advantage

$Adv_{\mathcal{A}}^{HASH}(t)$, we have $Adv2\ (et_2, q_R) \leq \epsilon_2$. As a result, our scheme is also secure against an adversary for deriving the secret key X_k of a physician server PS_k. □

Algorithm 2 EXP2

1: Eavesdrop the authentication request message $\langle ID_{MS_j}, ID_k, M_{12}, M_{13}, M_{14}, M_{15}, TS_{ms}\rangle$ during the authentication and session key agreement phase.
2: Call $Reveal$ oracle on input M_{15} to retrieve the information $(ID_i'||\ M_{11}'||\ M_{12}'||\ M_{13}'||\ M_{14}'||\ M_9'||\ R_{ms}'||\ TS_{ms}') \leftarrow Reveal(M_{15})$.
3: **if** $(TS_{ms}' = TS_{ms})$ and $(M_{12}' = M_{12})$ and $(M_{13}' = M_{13})$ and $(M_{14}' = M_{14})$
4: Compute $ID_i'' = M_{14} \oplus (M_9' \oplus R_{ms}')$.
5: **if** $(ID_i'' = ID_i')$
6: Compute $x = M_{13} \oplus R_{ms}'$.
7: Call $Reveal$ oracle on input x to retrieve the information ID_i and X_k as $(ID_i^*||X_k^*) \leftarrow Reveal(x)$.
8: **if** $(ID_i^* = ID_i')$
9: Accept X_k^* as the correct secret key X_k of the physician server PS_k.
10: **return** 1 (Success)
11: **else**
12: **return** 0 (Failure)
13: **end if**
14: **else**
15: **return** 0 (Failure)
16: **end if**
17: **else**
18: **return** 0 (Failure)
19: **end if**

Informal Security Analysis

In this section, we analyze the security of our proposed scheme informally to show that our scheme provides strong security protection on the relevant security attacks. In the following, we justify that our scheme has the ability to tolerate several known attacks.

Off-line Identity-password Guessing Attack

Suppose that an adversary $\mathcal{A}$ extracts all the information $\{\langle ID_{MS_j}, A_j, P_j\rangle | 1 \leq j \leq m + m',\ , e_i, f_i, \tau_i\}$ stored in the smart card SC_i of a legal user U_i, where $RPW_i = h(ID_i||K||PW_i)$, $A_j = h(ID_i||X_j) \oplus RPW_i$, $P_j = h(ID_{MS_j}||X_j) \oplus RPW_i$, $e_i = h(ID_i||\sigma_i) \oplus K$ and $f_i = h(ID_i||RPW_i||\sigma_i)$. Clearly the identity ID_i and password PW_i are protected by the one-way cryptographic hash function $h(\cdot)$ under the security parameters: the random number K as well as secret biometric key σ_i. Thus, guessing the identity-password pair without the knowledge of the user personal biometrics B_i is computationally infeasible problem. Moreover, in our scheme the user password PW_i does not appear in the communication messages and identity ID_i is protected using the long-term secret tokens. As a result, our scheme successfully prevents the off-line identity-password guessing attack.

Privileged-insider Attack

In the registration phase of our scheme, a user U_i sends the registration request message $\langle ID_i, RPW_i\rangle$ securely to the MRS, where $RPW_i = h(ID_i||K||PW_i)$. The password PW_i in the request message is protected by the one-way hash function $h(\cdot)$ under the security parameter K. Therefore, guessing PW_i from the request message without the knowledge of K is again a computationally hard problem. Thus, our scheme resits the privileged-insider attack.

Stolen Smart Card Attack

Assume that the smartcard SC_i of a legal user U_i is lost/stolen. An attacker $\mathcal{A}$ can then try to login to a server MS_j using the lost/stolen smartcard SC_i by guessing the valid user credentials $\langle ID_i, PW_i, B_i\rangle$. However, guessing valid user credentials from the information stored in the smartcard SC_i is computationally infeasible due to the hardness of guessing user personal biometrics B_i. Hence, our scheme is secure against the stolen/lost smartcard attack.

User-server Impersonation Attack

In this attack, an adversary may try to impersonate a user or server by intercepting the communication messages between them. In our scheme, the medical server MS_j authenticates a user U_i, and U_i and physician server PS_k mutually authenticate each other in order to establish a shared session key SK_{U_i,PS_k}. Therefore, the attacker cannot compute the valid messages to authenticate at the servers MS_j or PS_k even if he/she intercepts the communication messages between them. Thus, our scheme resists the user-server impersonation attack.

Known key Secrecy

In this attack, an adversary may try to derive the previous session key using the current compromised session key. However, in our scheme, the session key $SK_{U_i,PS_k} = h(ID_i||ID_k||R_c||R_k||h(ID_i||X_k)||TS_k)$ is computed using the one-time random secrets and these are protected by the one way hash function $h(\cdot)$. Thus, all the session

keys are computed independently, and as a result, computing previous session key using the known session key is computationally infeasible problem.

Session key Agreement and Verification

In our scheme, a user U_i and a physician server PS_k mutually authenticate each other to establish the shared session key SK_{U_i,PS_k}, which is already proved using the widely-accepted BAN logic. Finally, after mutually authentication both U_i and PS_k agree on a common session key $SK_{U_i,PS_k} = h(ID_i||ID_k||R_c||R_k||h(ID_i||X_k)||TS_k)$. Thus, our scheme provides the session key agreement and verification property.

Session key Discloser Attack

The session key $SK_{U_i,PS_k} = h(ID_i||\ ID_k||\ R_c||\ R_k||\ h(ID_i||\ X_k)\ ||TS_k)$ between a user U_i and a physician server PS_k depends on the one-time random secrets R_c and R_k, and the long-term shared secret key X_k between PS_k and MS_j. Thus, computing the session key SK_{U_i,PS_k} without these parameters is computationally hard problem. As a result, our scheme resists the session disclosure attack.

Strong Replay Attack Protection

In order to avoid the reply attack, we have used both the random nonces and time stamps. As discussed in Step $A2$ of the authentication and session key agreement phase, our scheme successfully uses the random nonces to avoid the immediate reply attack within the time interval. Thus, our scheme provides the message freshness property to avoid strong replay attack.

Strong Man-in-the-middle Attack Protection

Suppose an adversary $\mathcal{A}$ intercepts the message $\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c\rangle$ during the login phase, where $M_3 = ID_i \oplus M_2 = ID_i \oplus h(ID_{MS_j}||X_j)$, $X_j = h(ID_{MS_j}||X_c)$, X_c is the 1024-bit secret key of the MRS, $M_4 = ID_i \oplus M_1 \oplus R_c = ID_i \oplus h(ID_i||X_j) \oplus R_c$, $M_5 = h(M_1||M_3||M_4||R_c||TS_c)$ and $M_1 = h(ID_i||X_j)$. Further, assume that $\mathcal{A}$ generates a new random nonce R_c' and changes M_4 to $M_4' = M_4 \oplus R_c' = ID_i \oplus h(ID_i||X_j) \oplus (R_c \oplus R_c')$. However, $\mathcal{A}$ can not modify M_5 because it involves computation of $M_1 = h(ID_i||X_j)$, which needs the secret key X_c of the MRS, ID_i as well as R_c. Since X_j (subsequently, X_c) is protected by the one-way cryptographic hash function $h(\cdot)$, it is computationally infeasible problem for $\mathcal{A}$ to modify M_5. As a result, $\mathcal{A}$ does not have any ability to modify this message and send to MS_j. In a similar manner, $\mathcal{A}$ does not have also the ability modify other messages $\langle ID_{MS_j}, ID_k, M_{12}, M_{13}, M_{14}, M_{15}, TS_{ms}\rangle$ and $\langle ID_k, M_{22}, M_{23}, M_{24}, TS_k\rangle$ during the authentication and key agreement phase. Thus, our scheme protects strongly the man-in-the-middle attack.

No Encryption/decryption

As in Amin-Biswas's scheme [1], our scheme does not use any cryptographic symmetric-key cryptosystem. Our scheme is free from usage of symmetric-key encryption/decryption and uses only cryptographic hash function $h(\cdot)$ for authentication and session key agreement purpose. Thus, our scheme is efficient in computation.

Fast Error Detection

In our scheme, the user credentials verification is done locally by the smart card of a legal user U_i without contacting the medical server MS_j. Moreover, the password and biometric updates also take place locally. The adversary can not generate the valid message to create the denial of service to the valid registered user without the valid user credentials. Therefore, our scheme avoids extra computation and communication costs, and also the denial of service attack by immediately detecting the invalid user credentials.

No Verification Table

As in Amin-Biswas's scheme [1], our scheme does not require any password-verifier table. Thus, an attacker has no ability to get the secret information of the entities.

Simulation for Formal Security Verification using AVISPA Tool

In this section, we simulate our scheme for the formal security verification using the most widely-accepted and used tool, called AVISPA (Automated Validation of Internet Security Protocols and Applications) [2] to show that our scheme is secure against the replay and man-in-the-middle attacks.

AVISPA is a push-button tool for the automated validation of Internet security-sensitive protocols and applications, which basically provides a modular and expressive formal language for specifying protocols and their security properties, and integrates different back-ends that implement a variety of state-of-the-art automatic analysis techniques [2]. We use the widely-accepted AVISPA tool for our formal security verification [12–14, 41, 49, 50]. AVISPA implements four back-ends: On-the-fly Model-Checker (OFMC), Constraint Logic based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC)

```
role useri (Ui, MRS, MSj, PSk   : agent,
        SKuimrs : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: the initiator, the user Ui
played_by Ui
def=
 local State  : nat,
     RPWi, PWi, Bi, K, IDi, IDmsj,
     IDk, Aj, Pj, TSc, TSms:  text,
     TSk, Rc, Rms, Rk, M1, M2, M3,
     M4, M5, Xc, Xk : text,
     Gen, Rep : hash_func
 const user_msj_tsc,  user_msj_rc,
     msj_psk_tsms, msj_psk_rms,
     psk_user_tsk, psk_user_rk,
     s1, s2, s3, s4 : protocol_id
init  State := 0
transition
% User registration phase
1. State = 0   /\ RCV(start) =|>
  State' := 1 /\ RPWi' := H(IDi.K.PWi)
% Send the registration request message to MRS securely
          /\ SND({IDi.RPWi'}_SKuimrs)
          /\ secret({Xc}, s1, MRS)
          /\ secret({PWi, Bi, K}, s2, Ui)
          /\ secret({Xk}, s3, {MSj, PSk})
          /\ secret({IDi}, s4, {Ui,MSj,PSk})
% Receive the smart card from MRS securely
2. State = 1 /\ RCV({IDmsj.xor(H(IDi.H(IDmsj.Xc)),
            H(IDi.K.PWi)).xor(H(IDmsj.H(IDmsj.Xc)),
            H(IDi.K.PWi)).H.Gen.Rep}_SKuimrs) =|>
% Login phase
State' := 2 /\ TSc' := new()
        /\ Rc' := new()
        /\ M1' := H(IDi.H(IDmsj.Xc))
        /\ M3' := xor(IDi, xor(IDmsj.H(IDmsj.Xc)))
        /\ M4' := xor(IDi, xor(H(IDi.H(IDmsj.Xc)),Rc'))
        /\ M5' := H(M1'.M3'.M4'.Rc'.TSc')
% Send the login request message to MSj
        /\ SND(IDmsj.IDk.M3'.M4'.M5'.TSc')
% Ui has freshly generated the value TSc for MSj
        /\ witness(Ui, MSj, user_msj_tsc, TSc')
% Ui has freshly generated the value Rc for MSj
        /\ witness(Ui, MSj, user_msj_rc, Rc')
%Authentication and session key agreement phase
% Receive the authentication reply message from PSk
3. State = 2  /\ RCV(IDk.xor(IDi,xor(Rc',Rk')).
            xor(H(IDi.Xk),Rk').
            H(H(IDi.IDk.Rc'.Rk'.H(IDi.Xk).TSk').
            xor(IDi,xor(Rc',Rk')).
            xor(H(IDi.Xk),Rk').Rc'.Rk'.TSk').TSk') =|>
% Ui's acceptance of the value TSk generated for Ui by PSk
State' := 3  /\ request(PSk, Ui, psk_user_tsk, TSk')
% Ui's acceptance of the value Rk generated for Ui by PSk
         /\ request(PSk, Ui, psk_user_rk, Rk')
end role
```

Fig. 2 Role specification in HLPSL for the user U_i

and Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP). A static analysis is performed to check the executability of the protocol, and then the protocol and the intruder actions are compiled into an intermediate format (If), which is the starting point for the four automated protocol analysis techniques. The detailed descriptions of these back-ends are given in [2]. In AVISPA, the designed protocols need to be specified in HLPSL language [55]. HLPSL is based on roles: the basic roles represent each participant role, and composition roles represent the scenarios of basic roles. Each role is independent from the others, which gets some initial information by parameters, and then communicates with the other roles by channels. The intruder is always denoted by i in HLPSL and the intruder is always modeled using the Dolev-Yao model [19] (also described in the threat model used in this paper) with the possibility for the intruder to assume a legitimate role in a protocol run. The role system defines the number of sessions, and the number of principals and the basic roles. The output format (OF) is produced by using one of the four back-ends. When the analysis of a protocol has been successful (by finding an attack or not), the output describes precisely what is the result, and under what conditions it has been obtained. The detailed formats of the OF can be found in [55].

Specifying the Protocol

In our HLPSL implementation, we have four basic roles for the user U_i, the MRS, the medical server MS_j and the physician server PS_k. The specification in HLPSL language for the role of the initiator, the user U_i is given in Fig. 2. In this implementation, U_i first receives the start signal, changes its state value from 0 to 1, and then sends the user registration request message $\langle ID_i, RPW_i \rangle$ securely to the MRS with the $SND()$ operation. U_i then receives a smart card with the necessary information securely from the MRS by the $RCV()$ operation.

```
role mrs (Ui, MRS, MSj, PSk   : agent,
        SKuimrs : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: the MRS
played_by MRS
def=
 local State  : nat,
     RPWi, PWi, Bi, K, IDi, IDmsj,
     IDk, Aj, Pj, Xc, Xk  : text,
     Gen, Rep : hash_func
 const s1, s2, s3, s4 : protocol_id
init  State := 0
transition
% Registration phase
% Receive the registration request message from Ui
1. State  = 0 /\ RCV({IDi.H(IDi.K.PWi)}_SKuimrs) =|>
  State' := 1 /\ secret({Xc}, s1, MRS)
          /\ secret({PWi, Bi, K}, s2, Ui)
          /\ secret({Xk}, s3, {MSj, PSk})
          /\ secret({IDi}, s4, {Ui,MSj,PSk})
          /\ Aj' := xor(H(IDi.H(IDmsj.Xc)),
                H(IDi.K.PWi))
          /\ Pj' := xor(H(IDmsj.H(IDmsj.Xc)),
                H(IDi.K.PWi))
% Send the smart card to Ui securely
          /\ SND({IDmsj.Aj'.Pj'.H.Gen.Rep}_SKuimrs)
end role
```

Fig. 3 Role specification in HLPSL for the MRS

```
role medical_serverj (Ui, MRS, MSj, PSk : agent,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: MSj
played_by MSj
def=
 local State  : nat,
     RPWi, PWi, Bi, K, IDi, IDmsj,
     IDk, Xc, Xk, Aj, Pj, TSc, TSms:  text,
     TSk, Rc, Rms, Rk, M12, M13,
     M14, M15 : text,
     Gen, Rep : hash_func
 const user_msj_tsc,  user_msj_rc,
     msj_psk_tsms, msj_psk_rms,
     psk_user_tsk, psk_user_rk,
     s1, s2, s3, s4 : protocol_id
init  State := 0
transition
% Login phase
% Receive the login request message from Ui
1. State  = 0 ∧ RCV(IDmsj.IDk.
            xor(IDi, xor(IDmsj.H(IDmsj.Xc))).
            xor(IDi, xor(H(IDi.H(IDmsj.Xc)),Rc')).
            H(H(IDi.H(IDmsj.Xc)).
            xor(IDi, xor(IDmsj.H(IDmsj.Xc))).
            xor(IDi, xor(H(IDi.H(IDmsj.Xc)),Rc')).
            Rc'.TSc').TSc') =|>
  State' := 1 ∧ secret({Xc}, s1, MRS)
          ∧ secret({PWi, Bi, K}, s2, Ui)
          ∧ secret({Xk}, s3, {MSj, PSk})
          ∧ secret({IDi}, s4, {Ui,MSj,PSk})
% Authentication and session key agreement phase
          ∧ TSms' := new()
          ∧ Rms' := new()
          ∧ M12' := xor(IDi,H(IDmsj.IDk.Xk))
          ∧ M13' := xor(H(IDi.Xk),Rms')
          ∧ M14' := xor(IDi,xor(Rc',Rms'))
          ∧ M15' := H(IDi.H(IDmsj.IDk.Xk).M12'.M13'.
                M14'.Rc'.Rms'.TSms')
% Send authentication request message to PSk
          ∧ SND(IDmsj.IDk.M12'.M13'.M14'.M15'.TSms')
% MSj has freshly generated the value TSms for PSk
          ∧ witness(MSj, PSk, msj_psk_tsms, TSms')
% MSj has freshly generated the value Rms for PSk
          ∧ witness(MSj, PSk, msj_psk_rms, Rms')
% MSj's acceptance of the value TSc generated for MSj by Ui
          ∧ request(Ui, MSj, user_msj_tsc, TSc')
% MSj's acceptance of the value Rc generated for MSj by Ui
          ∧ request(Ui, MSj, user_msj_rc, Rc')
end role
```

Fig. 4 Role specification in HLPSL for the medical server MS_j

During the login phase, U_i sends the login request message $\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c \rangle$ to MS_j via a public channel. Finally, during the authentication and session key agreement phase, U_i receives the authentication reply message $\langle ID_k, M_{22}, M_{23}, M_{24}, TS_k \rangle$ from PS_k via a public channel. The type declaration $channel\,(dy)$ declares that the channel is for the Dolev-Yao threat model (as described in our threat model) [19]. The intruder (i) will have the ability to intercept, analyze, and/or modify messages transmitted over the insecure channel. The declaration secret(PWi, Bi, K, s2, Ui) means that the information PW_i, B_i and K are only known to the user U_i, which are characterized by the protocol id $s2$. The declaration witness(A, B, id, E) declares for a (weak) authentication property of A by B on E and declares that agent A is witness for the information E. This goal is identified by the constant id in the goal Section [2]. request(B, A, id, E) declaration means for a strong authentication property of A by B on E and declares that agent B requests a check of the value E. This goal is identified by the constant id in the goal section [2]. In a similar way, we have also specified the HLPSL implementation for the roles of the MRS, the medical server MS_j and the physician server PS_k in Fig. 3, Figs. 4 and 5, respectively.

In Fig. 6, we have shown the HLPSL implementation for the role of the session. In the session segment, all the basic roles: alice and bob are instanced with concrete arguments.

Finally, in Fig. 7, we have shown the HLPSL implementation for the role of goal and environment. The top-level role (called the environment) needs to be always defined in the specification of HLPSL language. This contains the global constants and a composition of one or more sessions, where the intruder may play some roles as legitimate users. The intruder (i) also participates in the

```
role physician_serverk (Ui, MRS, MSj, PSk : agent,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: PSk
played_by PSk
def=
 local State  : nat,
     RPWi, PWi, Bi, K, Xc, Xk, IDi,
     IDmsj, IDk, Aj, Pj, TSc, TSms:  text,
     TSk, Rc, Rms, Rk, M22,
     M23, M24 : text,
     Gen, Rep : hash_func
 const user_msj_tsc,  user_msj_rc,
     msj_psk_tsms, msj_psk_rms,
     psk_user_tsk, psk_user_rk,
     s1, s2, s3, s4 : protocol_id
init  State := 0
transition
% Authentication and session key agreement phase
% Receive authentication request message from MSj
1. State = 0 ∧ RCV(IDmsj.IDk.xor(IDi.
            H(IDmsj.IDk.Xk)).
            xor(H(IDi.Xk),Rms').
            xor(IDi,xor(Rc',Rms')).
       H(IDi.H(IDmsj.IDk.Xk).xor(IDi,H(IDmsj.IDk.Xk)).
       xor(H(IDi.Xk),Rms').
       xor(IDi,xor(Rc',Rms')).Rc'.Rms'.TSms').TSms') =|>
  State' := 1 ∧ secret({Xc}, s1, MRS)
          ∧ secret({PWi, Bi, K}, s2, Ui)
          ∧ secret({Xk}, s3, {MSj, PSk})
          ∧ secret({IDi}, s4, {Ui,MSj,PSk})
          ∧ TSk' := new()
          ∧ Rk' := new()
          ∧ M22' := xor(IDi,xor(Rc',Rk'))
          ∧ M23' := xor(H(IDi.Xk),Rk')
          ∧ M24' := H(H(IDi.IDk.Rc'.Rk'.H(IDi.Xk).TSk').
                M22'.M23'.Rc'.Rk'.TSk')
% Send authentication reply message to Ui
          ∧ SND(IDk.M22'.M23'.M24'.TSk')
% PSk has freshly generated the value TSk for Ui
          ∧ witness(PSk, Ui, psk_user_tsk, TSk')
% PSk has freshly generated the value Rk for Ui
          ∧ witness(PSk, Ui, psk_user_rk, Rk')
% PSk's acceptance of the value TSms generated for PSk by MSj
          ∧ request(MSj, PSk, msj_psk_tsms, TSms')
% PSk's acceptance of the value Rms generated for PSk by MSj
          ∧ request(MSj, PSk, msj_psk_rms, Rms')
end role
```

Fig. 5 Role specification in HLPSL for the physician server PS_k

```
role session (Ui, MRS, MSj, PSk : agent,
        SKuimrs : symmetric_key,
        % H is one−way hash function
        H: hash_func)
def=
 local  S1, S2, S3, S4, R1, R2, R3, R4: channel (dy)
 composition
        useri (Ui, MRS, MSj, PSk, SKuimrs, H, S1, R1)
      /\ mrs (Ui, MRS, MSj, PSk, SKuimrs, H, S2, R2)
      /\ medical_serverj (Ui, MRS, MSj, PSk, H, S3, R3)
      /\ physician_serverk (Ui, MRS, MSj, PSk, H, S4, R4)
end role
```

Fig. 6 Role specification in HLPSL for the session.

execution of protocol as a concrete session during the simulation as shown in this figure.

In our implementation, the following four secrecy goals and six authentication properties are verified:

- secrecy_of s1: It represents that X_c is kept secret to the MRS only.
- secrecy_of s2: It tells that PW_i, B_i and K are kept secret to the user U_i only.
- secrecy_of s3: In this case, X_k is kept secret to MS_j and PS_k.

```
role environment()
def=
  const ui, mrs, msj, psk: agent,
        skuimrs : symmetric_key,
        h, gen, rep   : hash_func,
        pwi, bi, xc, k, idi, idmsj, idk,
        rc, tsc, rms, tsms, rk, tsk:  text,
        user_msj_tsc,  user_msj_rc,
        msj_psk_tsms, msj_psk_rms,
        psk_user_tsk, psk_user_rk,
        s1, s2, s3, s4 : protocol_id
  intruder_knowledge = {ui, mrs, msj, psk, h,
                        idmsj, idk, gen, rep,
                        tsc, tsms, tsk}
  composition
   session(ui, mrs, msj, psk, skuimrs, h)
        /\ session(i, mrs, msj, psk, skuimrs, h)
        /\ session(ui, i, msj, psk, skuimrs, h)
        /\ session(ui, mrs, i, psk, skuimrs, h)
        /\ session(ui, mrs, msj, i, skuimrs, h)
end role

goal
  secrecy_of s1
  secrecy_of s2
  secrecy_of s3
  secrecy_of s4

  authentication_on user_msj_tsc
  authentication_on user_msj_rc
  authentication_on msj_psk_tsms
  authentication_on msj_psk_rms
  authentication_on psk_user_tsk
  authentication_on psk_user_rk
end goal
environment()
```

Fig. 7 Role specification in HLPSL for the goal and environment

- secrecy_of s4: It indicates that ID_i is known to U_i, MS_j and PS_k.
- authentication_on user_msj_tsc: When MS_j receives TS_c from the messages from U_i, MS_j authenticates U_i based on TS_c.
- authentication_on user_msj_rc: When MS_j receives R_c from the messages from U_i, MS_j also authenticates U_i based on R_c.
- authentication_on msj_psk_tsms: When PS_k receives TS_{ms} from the messages from MS_j, PS_k authenticates MS_j based on TS_{ms}.
- authentication_on msj_psk_rms: When PS_k receives R_{ms} from the messages from MS_j, PS_k also authenticates MS_j based on R_{ms}.
- authentication_on psk_user_tsk: When U_i receives TS_k from the messages from PS_k, U_i authenticates PS_k based on TS_k.
- authentication_on psk_user_rk: When U_i receives R_k from the messages from PS_k, U_i also authenticates PS_k based on R_k.

Analysis of the Results

We have simulated our scheme using the AVISPA web tool [3] for the widely-accepted OFMC back-end [4]. The simulation results for the formal security verification of our scheme using OFMC back-end are shown in Fig. 8. For replay attack check, OFMC backend verifies whether the legitimate agents can execute the specified protocol by performing a search of a passive intruder. OFMC backend then gives the intruder the knowledge of some normal sessions between the legitimate agents. The simulation results shown in Fig. 8 ensures that our scheme is secure against the replay

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/avispa/web−interface−computation/
  ./tempdir/workfile91NsmW.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 4.98s
  visitedNodes: 198 nodes
  depth: 8 plies
```

Fig. 8 The result of the analysis using OFMC backend

Table 7 Functionality comparison among our scheme and other existing related schemes

Functionality feature	[58]	[53]	[56]	[9]	[57]	[34]	[36]	[1]	Ours
A_1	Yes	Yes	No	Yes	No	No	Yes	Yes	Yes
A_2	No	No	No	Yes	No	No	Yes	No	Yes
A_3	Yes	Yes	No	No	No	No	Yes	Yes	Yes
A_4	No	Yes	No	No	No	No	Yes	Yes	Yes
A_5	Yes	No	No	Yes	Yes	No	Yes	No	Yes
$Skey$	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
MA	No	No	No	No	No	No	Yes	Yes	Yes
WPD	No	Yes	No	Yes	Yes	Yes	Yes	No	Yes
SKV	No	No	No	No	No	Yes	No	Yes	Yes
E/D	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes
SDF	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes

Note: A_1: whether resists off-line password guessing attack or not; A_2: whether resists insider attack or not; A_3: whether prevents user impersonation attack or not; A_4: whether resists session key discloser attack or not; A_5: whether resists replay attack or not; $Skey$: whether provides session key agreement or not; MA: whether satisfies mutual authentication or not; WPD: whether detects early wrong password or not; SKV: whether session key verification property is achieved or not; E/D: whether the protocol is independent of encryption/decryption algorithm or not; SDF: whether several design flaws are avoided or not.

attack. OFMC backend also checks if there is any man-in-the-middle attack possible by an intruder between the communications. In this backend, the depth for the search is 8, and 198 nodes have been searched in 4.98 s. It is thus evident from the results shown in Fig. 8 that our scheme also fulfills the design properties and it is secure under the test of AVISPA using OFMC backend with the bounded number of sessions. Hence, our scheme is secure against the passive attacks and the active attacks, such as the replay and man-in-the-middle attacks.

Performance Comparison with Other Related Schemes

In this section, we analyze the performance analysis of our scheme and compare with the related existing schemes, such as Yang-Yang's scheme [58], Sood et al.'s scheme [53], Wang-Ma's scheme [56], Chuang-Chen's scheme [9], Xue et al.'s scheme [57], Li et al.'s scheme [34], Maitra-Giri's scheme [36] and Amin-Biswas's scheme [1].

In Table 7, we have compared the functionality features among our scheme and other existing related schemes. From this table, it is evident that our scheme outperforms other schemes as our scheme supports extra features and efficient design in the login and authentication phases as compared to those for other existing schemes.

In Table 8, we have compared the computation costs required by our scheme and other existing schemes during the login and authentication phases. We have used the notations in this table as follows: T_h: execution time for one-way hash function; T_h: execution time for biohashing function; T_{fe} : execution time for a fuzzy extractor function $(Gen(\cdot)/Rep(\cdot))$; T_e: execution time for exponentiation

Table 8 Computation cost comparison among our scheme and existing related schemes

Scheme	Login phase	Authentication phase	Total cost
[58]	$4T_h + 1T_e$	$4T_e + 4T_h$	$8T_h + 5T_e$
[53]	$7T_h$	$24T_h$	$31T_h$
[56]	$4T_h + 2T_{spm}$	$7T_h + 4T_{spm}$	$11T_h + 6T_{spm}$
[9]	$4T_h$	$12T_h$	$16T_h$
[57]	$3T_h$	$24T_h$	$27T_h$
[34]	$2T_h$	$25T_h$	$27T_h$
[36]	$4T_h + 1T_e + 1T_{spm}$	$6T_h + 1T_{spm}$	$10T_h + 1T_e + 2T_{spm}$
[1]	$1T_H + 4T_h$	$14T_h$	$18T_h + 1T_H$
Ours	$1T_{fe} + 4T_h$	$14T_h$	$18T_h + 1T_{fe}$

Table 9 Communication cost comparison among our scheme and existing related schemes

Scheme	Communication cost in bits			Communication mode
	Login phase	Authentication phase	Total cost	
[58]	1472	1344	2816	(2) $SC \rightarrow S_j, S_j \rightarrow SC$
[53]	896	1216	2112	(5) $SC \rightarrow S_j, S_j \rightarrow CS, CS \rightarrow S_j, S_j \rightarrow SC, SC \rightarrow S_j$
[56]	320	256	576	(2) $SC \rightarrow S_j, S_j \rightarrow SC$
[9]	512	512	1024	(2) $SC \rightarrow S_j, S_j \rightarrow SC$
[57]	768	2176	2944	(4) $SC \rightarrow S_j, S_j \rightarrow CS, CS \rightarrow S_j, S_j \rightarrow SC$
[34]	512	1664	2176	(4) $SC \rightarrow S_j, S_j \rightarrow CS, CS \rightarrow S_j, S_j \rightarrow SC$
[36]	512	384	896	(2) $SC \rightarrow S_j, S_j \rightarrow SC$
[1]	768	1152	1920	(3) $SC \rightarrow MS, MS \rightarrow PS, PS \rightarrow SC$
Ours	480	1098	1578	(3) $SC \rightarrow MS, MS \rightarrow PS, PS \rightarrow SC$

Note: SC: Smart Card of a user U_i; S_j: Service provider server; CS: Control server; MS: Medical server; PS: Physician server.

operation; T_{spm}: execution time for encryption/decryption operation. Our scheme requires $1T_{fe} + 4T_h$ and $14T_h$ operations for the login phase, and authentication and session key agreement phase, respectively. On the other hands, Yang-Yang's scheme [58], Sood et al.'s scheme [53], Wang-Ma's scheme [56], Chuang-Chen's scheme [9], Xue et al.'s scheme [57], Li et al.'s scheme [34], Maitra-Giri's scheme [36] and Amin-Biswas's scheme [1] require $8T_h + 5T_e$, $31T_h$, $11T_h + 6T_{spm}$, $16T_h$, $27T_h$, $27T_h$, $10T_h + 1T_e + 2T_{spm}$ and $18T_h + 1T_H$ operations, respectively. Since the fuzzy extractor functions are efficient in computation, our scheme is also efficient in computation as compared to other schemes.

Finally, in Table 9, we have compared the communication costs required by our scheme and other existing schemes for the login and authentication phases. We assume that the bit lengths of identities ID_i, ID_{MS_j} and ID_k are 64 bits each, hash output and random number are of length 128 bits each, and timestamp is 32 bits. In this table, the third column (communication mode) represents the total number of messages required and the communication between the entities in the network. In our scheme, the communication costs required during the login phase for the message $\langle ID_{MS_j}, ID_k, M_3, M_4, M_5, TS_c \rangle$ is $(64 + 64 + 64 + 128 + 128 + 32) = 480$ bits, and during the authentication and session key agreement phase, the messages $\langle ID_{MS_j}, ID_k, M_{12}, M_{13}, M_{14}, M_{15}, TS_{ms} \rangle$ and $\langle ID_k, M_{22}, M_{23}, M_{24}, TS_k \rangle$ need $(64 + 64 + 64 + 128 + 128 + 128 + 32) + (64 + 128 + 128 + 128 + 32) = 1098$ bits. Thus, our scheme needs the total communication cost 1578 bits and requires 3 messages transmission. On the other hands, Yang-Yang's scheme [58], Sood et al.'s scheme [53], Wang-Ma's scheme [56], Chuang-Chen's scheme [9], Xue et al.'s scheme [57], Li et al.'s scheme [34], Maitra-Giri's scheme [36] and Amin-Biswas's scheme [1] require the communication costs of 2816, 2112, 576, 1024, 2944, 2176, 896 and 1920 bits, respectively. Our scheme is efficient in computation as compared to Yang-Yang's scheme [58], Sood et al.'s scheme [53], Xue et al.'s scheme [57], Li et al.'s scheme [34] and Amin-Biswas's scheme [1]. Though our scheme requires more communication overhead as compared to Wang-Ma's scheme [56], Chuang-Chen's scheme [9], Maitra-Giri's scheme [36], either those schemes are vulnerable to known attacks or they are inefficient and do not support extra important functionality features. As compared to Amin-Biswas's scheme [1], our scheme is highly efficient and provides more functionality features.

Conclusion

In this paper, we have proposed a robust user authentication with key agreement scheme in hierarchical multi-medical server architecture in TMIS in order to erase several security drawbacks and design flaws found in Amin-Biswas's scheme. From the performance and functionality analysis, it is evident that our scheme provides more features and is efficient in communication and computation as compared to Amin-Biswas's scheme and other related existing schemes. The rigorous formal security analysis using both BAN logic and random oracle, and informal security analysis show that our scheme has the ability to tolerate various known attacks. In addition, the simulation results using the widely-accepted AVISPA tool for the formal security verification reveal that our scheme is also secure against passive and active adversaries. Furthermore, our scheme supports dynamic medical server addition in the network after initial deployment, and password and biometric update phase correctly and effectively without further contacting the MRS. Thus, our scheme is very suitable for practical applications in TMIS.

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

Conflict of interests The authors declare that there is no conflict of interest.

References

1. Amin, R., and 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.
2. AVISPA: Automated Validation of Internet Security Protocols and Applications. http://www.avispa-project.org/. Accessed on January 2013.
3. AVISPA: AVISPA Web Tool. http://www.avispa-project.org/web-interface/expert.php/. Accessed on March 2015.
4. Basin, D., Modersheim, S., OFMC, L.V., A symbolic model checker for security protocols. *Int. J. Inf. Secur.* 4(3):181–208, 2005.
5. Burnett, A., Byrne, F., Dowling, T., Duffy, A., A Biometric Identity Based Signature Scheme. *Int. J. Netw. Secur.* 5(3):317–326, 2007.
6. Burrows, M., Abadi, M., Needham, R., A logic of authentication. *ACM Trans. Comput. Syst.* 8(1):18–36, 1990.
7. Chatterjee, S., and Das, A.K., An effective ECC-based user access control scheme with attribute-based encryption for wireless sensor networks. *Secur. Commun. Netw.* 8(9):1752–1771, 2015.
8. Chatterjee, S., Das, A.K., Sing, J.K., A novel and efficient user access control scheme for wireless body area sensor networks. *J. King Saud Univ.-Comput. Inf. Sci.* 26(2):181–201, 2014.
9. Chuang, M.-C., and Chen, M.C., An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics. *Expert Syst. Appl.* 41(4):1411–1418, 2014.
10. Chuang, Y.-H., and Tseng, Y.-M., An efficient dynamic group key agreement protocol for imbalanced wireless networks. *Int. J. Netw. Manag.* 20(4):167–180, 2010.
11. 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.
12. Das, A.K., A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks. *Peer-to-Peer Netw. Appl.* 1–22, 2014. doi:10.1007/s12083-014-0324-9.
13. Das, A.K., A secure and efficient user anonymity-preserving three-factor authentication protocol for large-scale distributed wireless sensor networks. *Wirel. Pers. Commun.* 1–28, 2015. doi:10.1007/s11277-015-2288-3.
14. Das, A.K., A secure user anonymity-preserving three-factor remote user authentication scheme for the telecare medicine information systems. *J. Med. Syst.* 39(3):1–20, 2015.
15. Das, A.K., and Goswami, A., A secure and efficient uniqueness-and-anonymity-preserving remote user authentication scheme for connected health care. *J. Med. Syst.* 37(3):1–16, 2013.
16. Das, A.K., Paul, N.R., Tripathy, L., Cryptanalysis and improvement of an access control in user hierarchy based on elliptic curve cryptosystem. *Inf. Sci.* 209(C):80–92, 2012.
17. Das, A.K., Sharma, P., Chatterjee, S., Sing, J.K., A dynamic password-based user authentication scheme for hierarchical wireless sensor networks. *J. Netw. Comput. Appl.* 35(5):1646–1656, 2012.
18. Dodis, Y., Reyzin, L., Smith, A., Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Proceedings of the Advances in Cryptology (Eurocrypt'04), Vol. 3027, pp. 523–540: LNCS, 2004.
19. Dolev, D., and Yao, A., On the security of public key protocols. *IEEE Trans. Inf. Theory* 29(2):198–208, 1983.
20. Guo, P., Wang, J., Geng, X.H., Kim, C.S., Kim, J.-U., A variable threshold-value authentication architecture for wireless mesh networks. *J. Internet Technol.* 15(6):929–936, 2014.
21. He, D., Kumar, N., Chen, J., Lee, C.-C., Chilamkurti, N., Yeo, S.-S., Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks. *Multimed. Syst.* 21(1):49–60, 2015.
22. He, D., Kumar, N., Chilamkurti, N. *A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks*: Information Sciences, 2015. doi:10.1016/j.ins.2015.02.010.
23. He, D., Kumar, N., Chilamkurti, N., Lee, J.-H., Lightweight ECC based RFID authentication integrated with an ID verifier transfer protocol. *J. Med. Syst.* 38(10), 2014.
24. He, D., Kumar, N., Lee, J.-H., Sherratt, R.S., Enhanced three-factor security protocol for consumer USB mass storage devices. *IEEE Trans. Consum. Electron.* 60(1):30–37, 2014.
25. He, D., and Zeadally, S., Authentication protocol for an ambient assisted living system. *IEEE Commun. Mag.* 53(1):71–77, 2015.
26. Islam, S.K.H., and Khan, M.K., Cryptanalysis and Improvement of Authentication and Key Agreement Protocols for Telecare Medicine Information Systems. *J. Med. Syst.* 38(10):135, 2014.
27. Jina, A.T.B., Linga, D.N.C., Biohashing, A.G., Two factor authentication featuring fingerprint data and tokenized random number. *Pattern Recogn.* 37(11):2245–2255, 2004.
28. Khan, M.K., and Kumari, S., An authentication scheme for secure access to healthcare services. *J. Med. Syst.* 37(4), 2013.
29. Khan, M.K., and Kumari, S., Cryptanalysis and Improvement of "An Efficient and Secure Dynamic ID-based Authentication Scheme for Telecare Medical Information Systems. *Secur. Commun. Netw.* 7(2):399–408, 2014.
30. Khan, M.K., and Kumari, S., An improved user authentication protocol for healthcare services via wireless medical sensor networks. *Int. J. Distrib. Sensor Netw.* 2014:1–10, 2014. doi:10.1155/2014/347169. Article ID 347169.
31. Kocher, P., Jaffe, J., Jun, B., Differential power analysis. In: Proceedings of Advances in Cryptology - CRYPTO'99, Vol. 1666, pp. 388–397: LNCS, 1999.
32. Kumari, S., Khan, M.K., Kumar, R., Cryptanalysis and improvement of 'a privacy enhanced scheme for telecare medical information systems. *J. Med. Syst.* 37(4), 2013.
33. Li, X., Niu, J.-W., Ma, J., Wang, W.-D., Liu, C.-L., Cryptanalysis and improvement of a biometrics-based remote user authentication scheme using smart cards. *J. Netw. Comput. Appl.* 34(1):73–79, 2011.
34. 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.
35. Lumini, A., and Nanni, L., An improved biohashing for human authentication. *Pattern Recogn.* 40(3):1057–1065, 2007.
36. Maitra, T., and Giri, D., An efficient biometric and password-based remote user authentication using smart card for telecare medical information systems in multi-server environment. *J. Med. Syst.* 38(12):1–19, 2014.
37. 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.

38. 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.
39. Mishra, D., On the security flaws in ID-based password authentication schemes for telecare medical information systems. *J. Med. Syst.* 39(1):154, 2014.
40. Mishra, D., Understanding security failures of two authentication and key agreement schemes for telecare medicine information systems. *J. Med. Syst.* 39(3):19, 2015.
41. 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.* 1–22, 2014. doi:10.1007/s12083-014-0321-z.
42. 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.
43. Mishra, D., and Mukhopadhyay, S., Cryptanalysis of Pairing-Free Identity-Based Authenticated Key Agreement Protocols. In: Information Systems Security, volume 8303 of Lecture Notes in Computer Science, pp. 247–254: Springer Berlin Heidelberg, 2013.
44. Mishra, D., Mukhopadhyay, S., Chaturvedi, A., Kumari, S., Khan, M.K., et al., Cryptanalysis and Improvement of Yan Biometric-Based Authentication Scheme for Telecare Medicine Information Systems. *J. Med. Syst.* 38(6):24, 2014.
45. Mishra, D., Mukhopadhyay, S., Kumari, S., Khan, M.K., Chaturvedi, A., Security enhancement of a biometric based authentication scheme for telecare medicine information systems with nonce. *J. Med. Syst.* 38(5):41, 2014.
46. Mishra, D., Srinivas, J., Mukhopadhyay, S., A secure and efficient chaotic map-based authenticated key agreement scheme for telecare medicine information systems. *J. Med. Syst.* 38(10), 2014.
47. Mishra, R., and Barnwal, A.K., A privacy preserving secure and efficient authentication scheme for telecare medical information systems. *J. Med. Syst.* 39(5):54, 2015.
48. Odelu, V., Das, A.K., Goswami, A., A secure and efficient ecc-based user anonymity preserving single sign-on scheme for distributed computer networks. *Secur. Commun. Netw.* 8(9):1732–1751, 2015.
49. Odelu, V., Das, A.K., Goswami, A. *A secure and scalable group access control scheme for wireless sensor networks*: Wireless Personal Communications, 2015. doi:10.1007/s11277-015-2866-4.
50. Odelu, V., Das, A.K., Goswami, A., A secure biometrics-based multi-server authentication protocol using smart cards. *IEEE Trans. Inf. Forensic. Secur.* 10(9):1953–1966, 2015. doi:10.1109/TIFS.2015.2439964.
51. Sarkar, P., A simple and generic construction of authenticated encryption with associated data. *ACM Trans. Inf. Syst. Secur.* 13(4):1–16, 2010.
52. Siddiqui, Z., Abdullah, A.-H., Khan, M.K., Alghamdi, A.S., Smart environment as a service, three factor cloud based user authentication for telecare medical information system. *J. Med. Syst.* 38(1):9997, 2014.
53. 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.
54. Stinson, D.R., Some Observations on the Theory of Cryptographic Hash Functions. *Des., Codes Crypt.* 38(2):259–277, 2006.
55. Von Oheimb, D., The high-level protocol specification language hlpsl developed in the eu project avispa, pp. 1–17: Tallinn, 2005.
56. Wang, B., and Ma, M., A smart card based efficient and secured multi-server authentication scheme. *Wirel. Pers. Commun.* 68(2):361–378, 2013.
57. Xue, K., Hong, P., Ma, C., A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture. *J. Comput. Syst. Sci.* 80(1):195–206, 2014.
58. Yang, D., and Yang, B., A biometric password-based multi-server authentication scheme with smart card. In: 2010 International Conference on Computer Design and Applications (ICCDA), Vol. 5, pp. 554–559: IEEE, 2010.