Wireless Pers Commun (2017) 95:2735–2767
DOI 10.1007/s11277-017-3956-2

A Multi-server Environment with Secure and Efficient Remote User Authentication Scheme Based on Dynamic ID Using Smart Cards

Srinivas Jangirala[1] · **Sourav Mukhopadhyay**[1] · **Ashok Kumar Das**[2]

Published online: 25 January 2017

Abstract The growth of the Internet and telecommunication technology has facilitated remote access. During the last decade, numerous remote user authentication schemes based on dynamic ID have been proposed for the multi-server environment using smart cards. Recently, Shunmuganathan et al. pointed out that Li et al.'s scheme is defenseless in resisting the password guessing attack, stolen smart card attack and forgery attack. Furthermore, they showed the poor repairability and no two-factor security in Li et al.'s scheme. To surmount these security disadvantages, Shunmuganathan et al. proposed a remote user authentication scheme using smart card for multi-server environment and claimed that their scheme is secure and efficient. In this paper, we show that Shunmuganathan et al.'s scheme is also defenseless in resisting the password guessing attack, stolen smart card attack, user impersonation attack, forgery attack, forward secrecy and session key secrecy. Moreover, the two-factor security is also not preserved in their scheme. In our proposed scheme, a user is free to choose his/her login credentials such as user id and password. And also a user can regenerate the password any time. Simultaneously the proposed scheme preserves the merits of Shunmuganathan et al.'s scheme and also provides better functionality and security features, such as mutual authentication, session key agreement and perfect forward secrecy. The security analysis using the widely accepted Burrows–Abadi–Needham logic shows that the proposed scheme provides the mutual authentication proof between a user and a server. Through the rigorous formal and informal security analysis, we show that the proposed scheme is secure against possible

✉ Srinivas Jangirala
jangiralasrinivas@maths.iitkgp.ernet.in

Sourav Mukhopadhyay
sourav@maths.iitkgp.ernet.in

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

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

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

known attacks. In addition, we carry out the simulation of the proposed scheme using the most-widely accepted and used Automated Validation of Internet Security Protocols and Applications tool and the simulation results clearly indicate that our scheme is secure.

Keywords Multi-server environment · Authentication · Anonymity · Session key · Smart card · BAN logic · AVISPA · Security

1 Introduction

The advances in network technologies make the remote intersection easier. It provides a scalable platform for numerous services over the public network such as e-learning, e-commerce, e-governance and e-medicine. Most of the services and applications are being provided over the public channels. The rapid development of the network and information technology have greatly improved the social production of the online services. The panoptic recurrence of the network applications fundamentally advances the improvement of multi-server architecture, in which the remote users have access to various distributed servers on the internet. The multi-server system consists of three participants, including the remote users, the servers and the registration center. The registration center (RC) acts as the trusted third party and it administrates all the registered servers and users. Service providing servers have complete control of the information services and the registered users are able to get these services. According to the number of participants in mutual authentication phase, there are two types of multi-server authentication protocols: one is implemented by two parties such as user and server, and the other is implemented by three parties such as user, registration center and server [27, 42].

The rapid development of the technology also given enough room for the information security problems. So, secure communication and data transfer between the participants has become essential requirement. In the process of providing secure communication many user authentication schemes based on identity have been proposed in the literature. To initiate the authentication process, at first the legitimacy of the user is verified by providing the login credentials.

In order to provide access to a user for any service, the service provider needs to verify the remote user's identity before giving access to the opted services. So, the identity authentication is needed and it is handy during verifying the validity of the user. Providing security for various types of online applications and services is the foremost barrier, which can restrict the unauthorized users from approaching service provider at various application systems. In recent days, a numerous number of identity based authentication schemes using smart card have been presented [8, 11–15, 17, 20–25, 39, 42, 47].

Keeping in view of basic cryptographic algorithms, the authentication schemes based on smart cards can be partitioned into two classifications, such as the hash based and the public key based authentication schemes. As per utilization of the methodology, the identity authentication schemes can be secluded into two: the single-server environment which can be relevant in user authentication mechanisms and the multi-server environment which can be suitable in user confirmation schemes, where the multi-server approval determines the issue existing in single-server accepts that the user needs to recollect various distinctive identities and passwords when he/she utilizes the single-server validation scheme to login and get to diverse remote servers. Subsequently, in recent years,

several smart card and identity based authentication schemes have been proposed [8, 11, 12, 17, 19, 23, 24, 39, 47, 49], and all of them have their own strengths and weaknesses. If the established password based authentication systems are employed in a multi-servers environment, a user does not just require to login to different remote servers. Additionally, for accessing various servers a remote user can avoid remembering various identities and passwords as it is not efficient and effectively arouses the compromise his/her personal identities and passwords. Note that one-time registration is an important problem in a multi-server environment as it significantly reduces the computational overhead and communication overhead caused by repeated registrations in order to access multiple servers.

In multi-server authentication schemes, the following criteria are considered to be important. Our proposed scheme has the ability to accomplish the following [3]:

C1: Select and modify the password at will
The user is free to select the password and modify their password whenever they wish in such a way that password can be memorized easily and remember by the user. This is essential in making the process flexible to the user and more user-friendly.

C2: Single registration
Earlier different remote servers offer various pairs of identities (IDs) and passwords (PWs) which confuses the user, while he/she registers at the remote system. On the other hand, memorizing all the identities and the corresponding passwords is not an easy task. In order to provide the best convenience, first the remote user needs to get register just only once at the registration center and thus, the legal user will be allowed to access the authorized servers.

C3: Security
The most important requirement in any authentication scheme is always considered to be secure. In real world, every authentication scheme should be able to restrict or prevent all kinds of attacks which are malicious.

C4: Mutual authentication
The authentication scheme should be able to provide mutual authentication and restrict any malicious attack.

C5: Session key agreement
Session key agreement is a very important in authentication schemes. The user and server should be able to handle and protect the session key for future secure communications.

C6: Low computation
Smart card has limited computational power. Thus, for more efficient and practical use, the user and the server should use lower computation load such that it can help in improving the efficiency.

1.1 Multi-server Environment

Figure 1 shows our proposed multi-server environment. The multi-server system consists of three participants: remote users, servers and the registration center. The registration center (the trusted third party) administrates all the registered servers and users. Service providing servers have complete control on the information services and the registered users are able to get these services. According to the number of participant in mutual authentication phase, there are two types of multi-server authentication protocols. One is

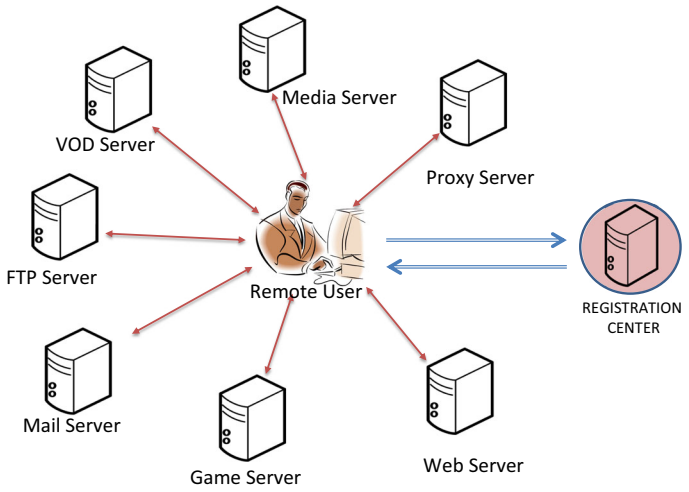

Fig. 1 The proposed multi-server environment

implemented by two parties: user and server, and the other is implemented by three parties: user, registration center and server [27, 42] . In this multi-server environment, each user should perform the registration procedure to register. The registered users can enjoy the services of the servers by logging to the servers. If the traditional authentication schemes are applied to this environment, the remote user needs to register with each and every server, and remembers the login credentials. Memorizing the login credentials of various servers is very much inconvenient and troublesome to the users. So, to overcome such trouble and facilitate the user with better login mechanism, several multi-server authentication proposals have been developed and presented [8, 13, 17, 20, 23, 24, 38, 39, 42, 47]. Each registered user gets the smart card from the registration center using which he/she can perform the login and authentication procedure to enjoy the services from the registered servers.

1.2 One-Way Collision-Resistant Hash Function

A one-way hash function h: $\{0,1\}^* \rightarrow \{0,1\}^k$ is considered as cryptographically secure and deterministic algorithm [5, 27, 36, 37, 41], which prefers input as an arbitrary length binary string $l \in \{0,1\}^*$ and produces an output as a binary string $h(l) \in \{0,1\}^k$ of a fixed-length k. The one way hash function may take the input in the form of a file, a message, or other blocks of data and also posses the listed properties, which are as follows [40]:

- The hash function $h(\cdot)$ can be performed on a data block of all sizes.
- The message digest $h(l)$ is easy to perform on any given input l.
- The output $h(l)$ produced by the hash function $h(\cdot)$ is fixed in length.
- Pre-image resistant: For a given hash value p, it is hard to find any l such that $p = h(l)$.
- Second Pre-image resistant: For a given l_1, it is computationally infeasible to find any l_2 with $l_1 \neq l_2$ such that $h(l_1) = h(l_2)$.
- Collision resistant: It is computationally infeasible to find any pair (l_1, l_2) with $l_1 \neq l_2$ such that $h(l_1) = h(l_2)$.

1.3 Threat Model

In this paper, we use the following threat model considering the assumptions as follows [9, 10, 16, 26, 48]:

- An adversary can extract the information from the smart card by examining the power consumption or leaked information.
- An adversary is able to eavesdrop all the communications between the parties involved such as a user and a server over a public channel.
- An adversary has the potential to modify, delete, redirect and resend the eavesdropped transmitted messages.
- An adversary can be a legal user or an outsider in any system.

1.4 Our Contributions

The contributions are listed below:

- We have analyzed the recently proposed Shunmuganathan et al.'s scheme and pointed out that their scheme suffers from several attacks such as stolen smart card attack, password guessing attack, impersonation attack, replay attack, forgery attack and forward secrecy attack.
- To overcome the security weaknesses of Shunmuganathan et al.'s scheme, we have proposed an efficient and more secure multi-server authentication scheme that can preserve all the original merits of Shunmuganathan et al.'s scheme and withstands the possible known attacks.
- To strengthen our proposed scheme, the security analysis using the BAN Logic has been presented. Using the informal security analysis, we have also shown our scheme can resist numerous security attacks which include the attacks found in Shunmuganathan et al.'s scheme.
- In addition, we have performed the simulation of the proposed scheme using the most-widely accepted and used Automated Validation of Internet Security Protocols and Applications (AVISPA) tool and the simulation results clearly indicate that our scheme is secure.
- Furthermore, our scheme is computationally efficient as compared to Shunmuganathan et al.'s scheme and other related existing schemes.

1.5 Notations

In Table 1, we have listed the notations which are used throughout the paper.

1.6 Organization of the Paper

The rest of this paper is arranged as follows. In Sect. 2, we present the related work in the literature. In Sect. 3, we present briefly Shunmuganathan et al.'s scheme. Subsequently, we demonstrate the weaknesses of Shunmuganathan et al.'s scheme in Sect. 4. In Sect. 5, we propose our scheme based on dynamic identity. To strengthen our proposed scheme, the security analysis using BAN Logic, and informal security analysis are provided in Sect. 6. We perform the simulation of the proposed scheme using the most-widely accepted and used AVISPA tool for the formal security verification in Sect. 7. In Sect. 8, the security

Table 1 Notations and their meanings

Symbol	Description
U_i	ith remote user
S_j	jth remote server
RC	Registration center
ID_i	Dynamic identity of the user U_i
SID_j	Identity of the server S_j
x	Master secret key of RC
y	Secret number generated by RC
b	Random number used by U_i
PW_i	Password of the user U_i
$SKey_{ij}$	Session key shared between U_i and S_j
$h(\cdot)$	Secure collision-free one-way hash function
$A\|\|B$	Concatenation of data A with data B
$A \oplus B$	Exclusive-OR of data A and data B
$\Rightarrow$	Participants communicating over a secure channel
$\rightarrow$	Participants communicating over a public channel

and performance comparisons of our scheme with various relevant schemes are shown. Finally, Sect. 9 concludes the paper.

2 Related Work

Lamport [18] was the first one who proposed password authentication concept to address the security issues over insecure public channels. Nevertheless, Lamport's scheme needs to store the user passwords. In 2000, based on the ElGamal's public key cryptosystem, Hwang and Li [25] proposed a user authentication scheme using smart cards, where the server does not need any storage of the password tables for authentication. Since then numerous researchers have focused on design and improvement of authentication schemes using one-way hash function for the single-server architecture [47]. However, when the single server architectures are connected to multi-server environment, the user faces various troubles such as repetitive registration to login to different remote servers and also he/she additionally needs to recollect different identities and passwords.

In 2004, based on symmetric key cryptosystem and hash function, Juang [14] proposed a multi-server authentication scheme, in which the public parameters require large storage memory. However, Chang and Lee [3] figured out that Juang's scheme is defenseless against dictionary attack as the secret values of the smart card can be extracted, and also demonstrated that Juang's scheme is inefficient. To overcome these pitfalls, Chang and Lee additionally proposed their enhanced remote user authentication scheme. But, later it was shown that their scheme is insecure against insider attack. In 2004, utilizing the Lagrange interpolation polynomial and RSA cryptosystem, Tsaur et al. [44] proposed a multi-server authentication scheme. Unfortunately, Tsaur et al.'s scheme is inefficient. In 2008, based on random nonce and hash function, Tsai [43] presented a multi-server authentication scheme. Their scheme takes the creditability as their scheme is designed without any verification table. Therefore, it is very much suitable for the distributed network environment with least computation expenses.

Regardless, these all displayed authentication schemes for multi-server environments have given a chance to aggressor to trace the legitimate user as the ID used by the user is static to the remote server during the login for every interaction session. Liao and Wang [24] proposed a multi-server authentication scheme based on dynamic identity, where a user presents his/her identity during the login phase to access the services from a remote server, which changes dynamically for each session. They claimed that their scheme can oppose numerous assaults and can accomplish mutual authentication. However, Hsiang and Shih [12] demonstrated that Liao and Wang's scheme is vulnerable to masquerade attack, insider attack, server and registration center spoofing attack. Further, they also proved that Liao and Wang's scheme does not facilitate mutual authentication. To beat these security pitfalls, Hsiang and Shih [12] presented an enhancement of Liao and Wang's authentication protocol for multi-server environment. Hsiang and Shih's scheme is also insecure against stolen smart card attack, impersonation attack and replay attack. Additionally, the password change phase is erroneous in their scheme. To resolve the security pitfalls in Hsiang and Shih's authentication scheme, Sood et al. [39] and Lee et al. [20] came up with authentication protocols using smart cards, which are based on dynamic identity for multi-server architecture.

Li et al. [23] encountered that Sood et al.'s scheme is helpless against leak-of-verifier attack, stolen smart card attack and impersonation attack. As a remedy, they proposed an efficient authentication protocol based on dynamic identity for multi-server architecture, which removes the aforementioned weaknesses of Sood et al.'s scheme [39]. Lee et al. [20] showed that Hsiang and Shih's scheme is suspectable to server spoofing attack and masquerade attack. Further, they showed that Hsiang and Shih's scheme does not provide mutual authentication. To avoid the security flaws of Hsiang and Shih's scheme, in 2011, Lee et al. [20] demonstrated an authentication scheme based on dynamic ID. Although they claimed that their scheme can withstand various attacks, Li et al. [23] showed that Lee et al.'s scheme can not furnish correct authentication and is also insecure against server spoofing attack and forgery attack. To overcome the shortcomings of Lee et al.'s scheme, Li et al. also demonstrated an authentication scheme, which is a dynamic ID based for multi-sever environment using smart cards.

Shunmuganathan et al. [38] pointed out the security flaws in Li et al.'s scheme and showed their scheme is inefficient to off-line password guessing attack, forgery attack and stolen smart card attack. And they also showed that Li et al.'s scheme faces poor reparability. To surmount these security disadvantages, Shunmuganathan et al. [38] proposed a scheme using smart card for multi-server environment and claims their scheme is secure and efficient. In this paper, we show that Shunmuganathan et al.'s scheme is also defenseless in resisting the off-line password guessing attack, stolen smart card attack, user impersonation attack, forgery attack, forward secrecy and session key secrecy. Their scheme does not also preserve the two-factor security. In our proposed scheme, a user is free to choose his/her login credentials such as user id and password and regenerate the password any time. In the meantime, the proposed scheme provides more functionality and security features such as session key agreement, perfect forward secrecy and mutual authentication.

3 Review of Shunmuganathan et al.'s Scheme

In this section, for a detailed investigation we review Shunmuganathan et al.'s remote user authentication scheme for multi-server environment based on dynamic ID using smart cards. In Shunmuganathan et al.'s scheme, we have three participants: the user (U_i), the

server (S_j), and the registration center (RC). The registration center (RC), selects x as the master secret key and y as a secret number to compute $h(x\|y)$ and $h(y)$, and then shares them with S_j over a secure channel. Their scheme consists of four phases: registration phase, login phase, verification phase, and password change phase. The detailed description of their scheme are described phase-wise below.

3.1 Registration Phase

The RC generates x as a master secret key and y as a secret number. The RC then computes $h(y)$ and $h(x\|y)$, and securely share them with S_j through a secure channel, when S_j is registered with RC. In order to enjoy the services of S_j, a new U_i must first register with the RC, where S_j has already been registered. This registration is a one-time activity, which can then be used to access all the servers S_j. U_i undergoes the registration procedure using the following steps:

R1: U_i creates an account in the multi-server system by freely choosing his/her identity ID_i, a random number b and password PW_i. Then, U_i computes $A_i = h(b\|PW_i)$ to hide the password PW_i. U_i sends ID_i and A_i to the RC via a secure channel for registration.

R2: After receiving ID_i and A_i from the user U_i securely, the RC computes $B_i = h(ID_i\|x)$, $C_i = h(ID_i\|h(y)\|A_i)$, $D_i = h(B_i\|h(x\|y)) \oplus A_i$ and $E_i = B_i \oplus h(x\|y)$.

R3: The RC issues a smart card containing the information $\{C_i, D_i, E_i, h(y), h(\cdot)\}$. The RC hands over the smart card to U_i through a secure channel.

R4: U_i keys b into his/her smart card. Finally, the smart card contains the information $\{C_i, D_i, E_i, b, h(y), h(\cdot)\}$.

3.2 Login Phase

The smart card received from the RC through the registration phase is used to access S_j by a user U_i. Whenever U_i wants to login to S_j, U_i must generate a login request message using the following steps:

L1: U_i first places his/her smart card into the smart card reader and keys in ID_i and PW_i. The smart card computes $A_i = h(b\|PW_i)$ and $C_i^* = h(ID_i\|h(y)\|A_i)$ using the input values and the data into its memory. Then, it checks if the computed C_i^* is equal to C_i, which is stored in smart card. The equality lets the smart card validate the user U_i and proceed with the next step. Otherwise, the smart card aborts the session.

L2: The user's smart card generates two random nonces N_i and N_k, and computes $P_{ij} = E_i \oplus h(h(SID_j\|h(y))\|N_i)$, $F_i = D_i \oplus A_i$, $A1_i = h(A_i\|N_k)$, $CID_i = A1_i \oplus h(F_i \|SID_j\|N_i)$, $M_1 = h(P_{ij}\|CID_i\|F_i\|N_i)$.

L3: Finally, U_i's smart card sends $\{P_{ij}, CID_i, M_1, N_i\}$ as a login request message to S_j through a public channel.

3.3 Verification Phase

Once S_j receives the login request message $\{P_{ij}, CID_i, M_1, N_i\}$ from U_i, it performs the following steps to validate the legitimacy of the user U_i and complete mutual authentication:

V1: S_j computes the following using the login request message $\{P_{ij}, CID_i, M_1, N_i\}$ received from U_i, and $h(SID_j||h(y))$ and $h(x||y)$, which are known to S_j: $E_i = P_{ij} \oplus h(h(SID_j||h(y))||N_i)$, $B_i = E_i \oplus h(x||y)$, $F_i = h(B_i||h(x||y))$, $A1_i = CID_i \oplus h(F_i||SID_j||N_i)$.

V2: S_j computes $h(P_{ij}||CID_i||F_i||N_i)$ and checks whether this value is equal to M_1. If they are not equal, the request is not valid. S_j then declines the login request and terminates this session. Otherwise, it is a valid login request and S_j accepts the login request message and generates a nonce N_j, and computes $M_2 = h(F_i||A1_i||N_j||SID_j)$, $M_3 = A1_i \oplus N_i \oplus N_j$. S_j sends the message $\{M_2, M_3\}$ to U_i through a public channel for mutual authentication.

V3: U_i extracts N_j from the received message $\{M_2, M_3\}$ as $N_j = A1_i \oplus N_i \oplus M_3$ and computes $h(F_i||A1_i||N_j||SID_j)$ to check the equality with the received message M_2. If the condition is not satisfied, U_i declines further process and terminates the session. Otherwise, U_i successfully authenticates S_j, and computes the mutual authentication message $M_4 = h(F_i||A1_i||N_i||SID_j)$ and sends $\{M_4\}$ to the server S_j through a public channel.

V4: Upon receiving the message $\{M_4\}$ from U_i, S_j computes $h(F_i||A1_i||N_i||SID_j)$ and compares it with M_4. If the verification does not hold, S_j terminates the session. Otherwise, S_j is successful in authenticating U_i and thereby, the mutual authentication process is completed. Once the mutual authentication succeeds, U_i and S_j compute the session key for their future secure communication as $SK = h(F_i||A1_i||N_i||N_j||SID_j)$.

The login and verification phases of Shunmuganathan et al.'s scheme is summarized in Table 2.

3.4 Password Change Phase

U_i invokes this phase to change/modify the password PW_i to a new password PW_i^{new}. This phase does not require any secure channel or any interaction with the *RC*. The steps involved in this phase are as follows.

P1: U_i places the smart card into the smart card reader and keys in ID_i and PW_i.

P2: The smart card computes $A_i = h(b||PW_i)$ and $C_i^* = h(ID_i||h(y)||A_i)$. It then checks whether the computed C_i^* is equal to C_i. If the verification does not hold, the smart card declines the password change request. Otherwise, U_i is allowed to input a new password PW_i^{new} and a new random number b^{new}.

P3: The smart card computes $A_i^{new} = h(b^{new}||\ PW_i^{new})$, $C_i^{new} = h(ID_i||\ h(y)||\ A_1^{new})$ and $D_i^{new} = D_i \oplus A_i \oplus A_i^{new}$.

P4: Finally, the smart card replaces C_i with C_i^{new} and D_i with D_i^{new} to complete the password change phase.

4 Cryptanalysis on Shunmuganathan et al.'s Scheme

In this section, we analyze the security of the recently proposed Shunmuganathan et al.'s scheme and show that their scheme is vulnerable to the following attacks.

Table 2 Login and verification phases of Shunmuganathan et al.'s scheme

User U_i		Server S_i
Inputs ID_i, PW_i.		
Computes $A_i = h(b\|\|PW_i)$,		
$C_i^* = h(ID_i\|\|h(y)\|\|A_i)$.		
Checks $C_i^* \stackrel{?}{=} C_i$		
Generates two nonce N_i and N_k.		
Computes		
$P_{ij} = E_i \oplus h(h(SID_j\|\|h(y))\|\|N_i)$,		
$F_i = D_i \oplus A_i$,		
$A1_i = h(A_i\|\|N_k)$,		
$CID_i = A1_i \oplus h(F_i\|\|SID_j\|\|N_i)$,		
$M_1 = h(P_{ij}\|\|CID_i\|\|F_i\|\|N_i)$.		
	$\xrightarrow{\{P_{ij}, CID_i, M_1, N_i\}}$	
		Computes
		$E_i = P_{ij} \oplus h(h(SID_j\|\|h(y))\|\|N_i)$,
		$B_i = E_i \oplus h(x\|\|y)$,
		$F_i = h(B_i\|\|h(x\|\|y))$,
		$A1_i = CID_i \oplus h(F_i\|\|SID_j\|\|N_i)$.
		Checks $h(P_{ij}\|\|CID_i\|\|F_i\|\|N_i) \stackrel{?}{=} M_1$
		Generate a nonce N_j.
		Computes
		$M_2 = h(F_i\|\|A1_i\|\|N_j\|\|SID_j)$,
		$M_3 = A1_i \oplus N_i \oplus N_j$.
	$\xleftarrow{\{M_2, M_3\}}$	
Computes $N_j = A1_i \oplus N_i \oplus M_3$.		
Checks $h(F_i\|\|A1_i\|\|N_j\|\|SID_j) \stackrel{?}{=} M_2$		
Computes		
$M_4 = h(F_i\|\|A1_i\|\|N_i\|\|SID_j)$.		
	$\xrightarrow{\{M_4\}}$	
		Checks $h(F_i\|\|A1_i\|\|N_i\|\|SID_j) \stackrel{?}{=} M_4$
Shared session key is $SK = h(F_i\|\|A1_i\|\|N_i\|\|N_j\|\|SID_j)$.		

4.1 Password Guessing Attack

We assume that the user U_i's smart card is lost or stolen. An attacker can then extract the information $\{C_i, D_i, b, E_i, h(y), h(\cdot)\}$, which are stored in the smart card according to the threat model provided in Sect. 1.3. In this attack, an attacker tries to guess the password PW_i by computing $A_i^* = h(b\|\|PW_i')$, $F_i^* = A_i^* \oplus D_i$, $M_1^* = h(P_{ij}\|\|CID_i\|\|F_i^*\|\|N_i)$, and then checking $M_1^* \stackrel{?}{=} M_1$. If the verification is successful, the attacker guesses the correct password. Otherwise, the attacker repeats this process until he/she obtains the correct password.

This shows that Shunmuganathan et al.'s scheme is insecure against the password guessing attack.

4.2 User Impersonation Attack

If F_i is known, the user impersonation attack is possible in Shunmuganathan et al.'s scheme. This is achieved while the password guessing attack is successful. The attacker computes $F_i^* = A_i^* \oplus D_i$, and $M_1^* = h(P_{ij}\|CID_i^*\|F_i^*\|N_i)$ and then transmits the message $\{P_{ij}, CID_i^*, M_1^*, N_i\}$ to S_j. S_j verifies M_1^* by computing $E_i = P_{ij} \oplus h(h(SID_j\|h(y))\|N_i)$, $B_i = E_i \oplus h(x\|y)$, $F_i = h(B_i\|h(x\|y))$, $A1_i = CID_i^* \oplus h(F_i\|SID_j\|N_i)$, and then checking the condition $h(P_{ij}\|CID_i^*\|F_i\|N_i) \stackrel{?}{=} M_1^*$. This verification gets successful. Hence, the attacker can successfully impersonate the user.

4.3 Stolen Smartcard Attack

An adversary can successfully login into the system with the extracted information $\{C_i, D_i, E_i, b, h(y), h()\}$ from the smart card and guessing a correct password PW_i^* as explained in Sect. 4.1. Furthermore, in Sect. 4.2, we have shown using the stolen smart card, an adversary can easily impersonate a legitimate user U_i even without knowing the valid ID_i, and secret keys x and y. Thus, Shunmuganathan et al.'s scheme is insecure against the stolen smart card attack.

4.4 No Two-Factor Security

Shunmuganathan et al.'s scheme scheme completely violates the two-factor security due to the following reason. First, our analysis under the off-line password guessing attack proves that their scheme does not provide two-factor authentication. Next, our analysis under stolen smart card attack shows that once the data in the smart card are extracted, a valid login message can be created even without knowing ID_i of the user U_i. An adversary can also login any number of times without being detected by S_j. S_j cannot prevent the further misuse of the smart card as it has no provision to securely revoke the stolen smart card.

4.5 Forgery Attack

The discussed stolen smart card attack (Sect. 4.3) proves that an adversary can forge as a legal user to login to the remote server S_j. The adversary can forge a valid login request $\{P_{ij}, CID_i^*, M_1^*, N_i\}$ using the data stored in the stolen smart card to fool S_j. The adversary can also manipulate the login request just with the data stored in the smart card without even knowing ID_i as explained in the stolen smart card and user impersonation attacks. This is clear that Shunmuganathan et al.'s scheme cannot withstand the forgery attack.

4.6 Good Repairability

A good login and authentication mechanism should be able to revoke lost or stolen smart card with good repairability. The registration center RC should be able to revoke the smart card, if at all it is lost or stolen. Although, Shunmuganathan et al. claims that their scheme provides good repairability, we have shown that their scheme is insecure against

stolen smart card and password guessing attack (Sects. 4.1 and 4.3). Therefore, Shunmuganathan et al.'s scheme does not provide good repairability.

4.7 Replay Attack

In this type of attack, an adversary tries to eavesdrop a valid login message $\{P_{ij}, CID_i^*, M_1^*, N_i\}$ between U_i and S_j. Then the adversary replays it back to S_j and tries to gain access to the service provided by S_j. After receiving a mutual authentication challenge response message $\{M_2, M_3\}$ from S_j, the adversary tries to generate a response message $\{M_4\}$ by extracting N_j from M_3. As we have already shown that the adversary can successfully compute M_4 and get $F_i, A1_i, N_i$. Therefore, the adversary can successfully generate M_4 and send it to S_j. Hence, Shunmuganathan et al.'s scheme cannot resist replay attack.

4.8 Forward Secrecy

From the above discussions in Sects. 4.5 and 4.7, an adversary can compute $F_i, A1_i$, and N_i. Also, from the transmitted message $\{M_2, M_3\}$ from S_j, the adversary can compute $N_j = A1_i \oplus N_i \oplus M3$. In order to compute a session key $SK = h(F_i \| A1_i \| N_i \| N_j \| SID_j)$, the adversary has all the required credentials. Hence, the adversary can easily compute the session key. As a result, Shunmuganathan et al.'s scheme does not preserve forward secrecy property.

5 Our Proposed Scheme

In this section, we present our proposed scheme, which has three participants, such as the registration center RC, the user U_i, and the server S_j. Our scheme possesses four phases, namely registration, login, authentication and key agreement, and password change phase. We use the notations listed in Table 1 for describing our scheme.

5.1 Registration Phase

The registration of U_i with the RC is a one-time activity. This phase consists of the following steps:

R1: U_i freely chooses his/her identity ID_i, password PW_i, and generates a random number b to compute $A_i = h(ID_i \oplus b \oplus PW_i)$. U_i then transmits ID_i and A_i to the RC for the registration purpose via a secure channel.

R2: The RC computes $B_i = h(A_i \| x)$, $C_i = h(ID_i \| h(y) \| A_i)$, $D_i = h(B_i \| h(x \| y))$, and $E_i = B_i \oplus h(x \| y)$. The RC stores $\{C_i, D_i, E_i, h(y), h(\cdot)\}$ on the user U_i's smart card and sends it to U_i via a secure channel.

R3: After receiving the smart card from the RC, U_i computes $L_i = b \oplus h(ID_i \| PW_i)$ and keys L_i into the smart card. Finally, the smart card contains the information $\{C_i, D_i, E_i, L_i, h(y), h(\cdot)\}$. Note that in our scheme, the random secret number b is not stored directly as compared to Shunmuganathan et al.'s scheme and instead of storing b the smart card stores L_i. This helps to prevent the privileged-insider attack in our scheme.

The user registration phase of our scheme is summarized in Table 3.

5.2 Login Phase

This phase helps U_i to send a login request message to S_j with the following steps:

L1: U_i places the smart card into the smart card reader, and inputs ID_i and PW_i.
L2: The smart card performs computations $b = L_i \oplus h(ID_i \| PW_i)$, $A_i = h(ID_i \oplus b \oplus PW_i)$ and $C_i^* = h(ID_i \| h(y) \| A_i)$ and then checks if the computed C_i^* is equal to C_i, which is available in the smart card. If there is a match, the smart card proceeds with the next step. Otherwise, the smart card aborts the session.
L3: The smart card generates a nonce N_i. The smart card then computes $CID_i = A_i \oplus h(D_i \| SID_j \| N_i)$, $P_{ij} = E_i \oplus h(h(SID_j \| h(y)) \| N_i)$, $M_1 = h(P_{ij} \| CID_i \| A_i \| N_i)$, and $M_2 = h(SID_j \| h(y)) \oplus N_i$.
L4: U_i finally sends the login request message $\{CID_i, P_{ij}, M_1, M_2\}$ to S_j via a public channel.

5.3 Authentication and Key Agreement Phase

In this phase, S_j and U_i execute the following steps to verify the login request and complete challenge response message for mutual authentication. At the end of this phase, both U_i and S_j agree on a secret session key $SKey_{ij}$ for their future secure communication.

V1: S_j computes $N_i = h(SID_j \| h(y)) \oplus M_2$, $E_i = P_{ij} \oplus h(h(SID_j \| h(y)) \| N_i)$, $B_i = E_i \oplus h(x \| y)$, $D_i = h(B_i \| h(x \| y))$, $A_i = CID_i \oplus h(D_i \| SID_j \| N_i)$ using the transmitted message credentials $\{P_{ij}, CID_i, M_1, M_2\}$ and known credentials $h(x \| y), h(y)$.
V2: S_j computes $h(P_{ij} \| CID_i \| A_i \| N_i)$ and checks whether it matches with M_1. If it does not hold, S_j declines the login request and terminates the session. Otherwise, S_j accepts the login request message and generates a nonce N_j. S_j then computes $SK_{ij} = h(h(B_i \| h(x \| y)) \| A_i)$, $M_3 = h(SK_{ij} \| A_i \| SID_j \| N_j)$ and $M_4 = SK_{ij} \oplus N_j$. S_j sends the message $\{M_3, M_4\}$ to U_i via a public channel.
V3: Upon receiving the message $\{M_3, M_4\}$ from S_j, U_i computes $SK_{ij} = h(D_i \| A_i)$, extracts N_j by computing $N_j = SK_{ij} \oplus M_4$, and checks whether $h(SK_{ij} \| A_i \| SID_j \| N_j)$

Table 3 User registration phase of our scheme

User U_i		RC
Computes $A_i = h(ID_i \oplus b \oplus PW_i)$.		
	$\xrightarrow{\{ID_i, A_i\}}$	Computes $B_i = h(A_i \| x)$,
		$C_i = h(ID_i \| A_i \| h(y))$,
		$D_i = h(B_i \| h(x \| y))$,
		$E_i = h(x \| y) \oplus B_i$.
	$\xleftarrow{Smartcard\langle C_i, D_i, E_i, h(y), h(\cdot)\rangle}$	
Computes $L_i = b \oplus h(ID_i \| PW_i)$.		
Inputs L_i in the smart card.		

is equal to M_3. If they are equal, U_i successfully authenticates S_j. U_i further computes $M_5 = h(SK_{ij}\|A_i\|SID_j\|N_i\|N_j)$ and sends the message $\{M_5\}$ to S_j via a public channel. Otherwise, the session is terminated.

V4: S_j computes $h(SK_{ij}\|A_i\|SID_j\|N_i\|N_j)$ and compares it with the received M_5. If they are equal, S_j successfully authenticates U_i and the mutual authentication is complete. Otherwise, the session is terminated. Then, both U_i and S_j compute a common session key $SKey_{ij} = h(SK_{ij}\|A_i\|SID_j\|N_i\|D_i\|N_j)$ for their secure future communication.

The login, and authentication and key agreement phases of our scheme are summarized in Table 4.

5.4 Password Change Phase

To change the old password PW_i to a new password PW_i^{new}, a user U_i needs to invoke this phase. There is no need to communicate with the *RC* further for the password change. This phase has the following steps:

P1: U_i places his/her smart card into the smart card reader, and then inputs ID_i and PW_i.

P2: The smart card computes $b^* = L_i \oplus h(ID_i\|PW_i)$, $A_i^* = h(ID_i \oplus b^* \oplus PW_i)$ and $C_i^* = h(ID_i\|h(y)\|A_i^*)$. The smart card then checks if the computed C_i^* is equal to C_i. If the equality does not hold, the smart card declines the request for password change. Otherwise, U_i keys in a new password PW_i^{new}.

P3: The smart card computes $A_i^{new} = h(ID_i \oplus b^* \oplus PW_i^{new})$ and $C_i^{new} = h(ID_i\|A_i^{new}\| h(y))$, and then replaces C_i with C_i^{new} .

P4: Finally, the smart card computes $L_i^{new} = b^* \oplus h(ID_i\|PW_i^{new})$ and replaces L_i with L_i^{new}.

6 Security Analysis of the Proposed Scheme

In this section, we show that the proposed scheme provides secure mutual authentication between a user U_i and a server S_j through the formal protocol analysis using the well-known widely-accepted BAN logic [2]. In addition, we simulate the proposed scheme for the formal security verification using the widely-accepted Automated Validation of Internet Security Protocols and Applications (AVISPA) tool [1] in Sect. 7 to show that it is secure against the replay and man-in-the-middle attacks. Wang et al. [46] pointed out that anyone of the formal security analysis, informal security analysis and AVISPA simulation analysis can not capture all the attacks. Hence, it is necessary for the proposed scheme to show that it is secure against possible known attacks using all possible security analysis. For this purpose, we also perform the informal security analysis to show that our scheme is also secure against other possible known attacks.

6.1 Authentication Proof Based on BAN Logic

The BAN logic [2] is widely being used to verify the correctness of the authentication protocol with key agreement. The protocol correctness refers to the communication parties U_i and S_j, who share a fresh shared session key with each other after the protocol is executed. We first provide some notations of the BAN logic as follows:

Table 4 Login, and authentication and key agreement phases of our scheme

User U_i		Server S_j
Inputs ID_i, PW_i.		
Computes $b = L_i \oplus h(ID_i \| PW_i)$,		
$A_i = h(ID_i \oplus b \oplus PW_i)$,		
$C_i^* = h(ID_i \| h(y) \| A_i)$.		
Checks $C_i^* \stackrel{?}{=} C_i$		
Accept/reject?		
Generates a random nonce N_i.		
Computes		
$CID_i = A_i \oplus h(D_i \| SID_j \| N_i)$,		
$P_{ij} = E_i \oplus h(h(SID_j \| h(y)) \| N_i)$,		
$M_1 = h(P_{ij} \| CID_i \| A_i \| N_i)$,		
$M_2 = h(SID_j \| h(y)) \oplus N_i$.		
	$\xrightarrow{\{CID_i, P_{ij}, M_1, M_2\}}$	
		Computes
		$N_i = h(SID_j \| h(y)) \oplus M_2$
		$E_i = P_{ij} \oplus h(h(SID_j \| h(y)) \| N_i)$,
		$B_i = E_i \oplus h(x \| y)$,
		$D_i = h(B_i \| h(x \| y))$,
		$A_i = CID_i \oplus h(D_i \| SID_j \| N_i)$.
		Checks
		$h(P_{ij} \| CID_i \| A_i \| N_i) \stackrel{?}{=} M_1$
		Accept/reject?
		Generates a random nonce N_j.
		Computes $SK_{ij} = h(h(B_i \| h(x \| y)) \| A_i)$,
		$M_3 = h(SK_{ij} \| A_i \| SID_j \| N_j)$,
		$M_4 = SK_{ij} \oplus N_j$.
	$\xleftarrow{\{M_3, M_4\}}$	
Computes $SK_{ij} = h(D_i \| A_i)$,		
$N_j = SK_{ij} \oplus M_4$,		
Checks		
$h(SK_{ij} \| A_i \| SID_j \| N_j) \stackrel{?}{=} M_3$		
Accept/reject?		
$M_5 = h(SK_{ij} \| A_i \| SID_j \| N_i \| N_j)$		
	$\xrightarrow{\{M_5\}}$	
		Checks
		$h(SK_{ij} \| A_i \| SID_j \| N_i \| N_j) \stackrel{?}{=} M_5$
		Accept/reject?
Shared session key is $SKey_{ij} = h(SK_{ij} \| A_i \| SID_j \| N_i \| D_i \| N_j$.		

- $P|\equiv X$: The principal P believes the announcement X.
- $P \triangleleft X$: P considers X, which means that a message containing X is received by P where X can be read by P.
- $P|\sim X$: P sometime stated X, which means that $P|\equiv X$: as P once stated it in sometime.
- $P|\Rightarrow X$:P commands X, P has complete authority on X, and P considers X as trusted (Jurisdiction over X).
- $\sharp(X)$: The message X is fresh, which means that no any entity sent a message containing X at whenever ahead of current round.
- $P|\equiv Q \overset{SK}{\longleftrightarrow} P$: P and Q use SK (shared key) to communicate with each other.
- $P \overset{SK}{\longleftrightarrow} Q$: P and Q use SK as a shared secret between them.
- $\{X\}_k$: The formula X is encrypted under the key k.
- $<X>_Y$: The formula X is combined with the formula Y.
- $(X)_k$: The formula X is hashed with the key k.

In order to describe logical postulates of BAN logic in formal terms[2], we present the following rules:

Rule (1). **Message meaning rule:** For shared secret keys:

$$\frac{P|\equiv Q \overset{k}{\longleftrightarrow} P, P \triangleleft \{X\}_k}{P|\equiv Q \sim X}. \tag{1}$$

P is said to believe Q, if P believes that k is shared with Q and P sees X encrypted under k.

Rule (2). **Nonce verification rule:**

$$\frac{P|\equiv \sharp(X), P|\equiv Q|\sim X}{P|\equiv Q|\equiv X}. \tag{2}$$

If P believes that X is expressed recently (freshness) and P believes that Q once said X, P believes that Q believes X.

Rule (3). **Jurisdiction rule:**

$$\frac{P|\equiv Q|\equiv X, P|\equiv Q|\Rightarrow X}{P|\equiv X}. \tag{3}$$

If P believes that Q has jurisdiction over X, and P believes that Q believes a message X, P believes X.

Rule (4). **Freshness rule:**

$$\frac{P|\equiv \sharp(X)}{P|\equiv \sharp(X,Y)}. \tag{4}$$

If one part known to be fresh, the entire formula must be fresh.

Rule (5). **Belief rule:**

$$\frac{P|\equiv Q|\equiv (X,Y)}{P|\equiv Q|\equiv (X)}. \tag{5}$$

If P believes Q believes the message set (X, Y), P also believes Q believes the message X.

According to the analytic procedures of the BAN logic, our proposed protocol should satisfy the following goals:

Goal 1. $U_i|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$;

Goal 2. $U_i|\equiv S_j|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$;

Goal 3. $S_j|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$;

Goal 4. $S_j|\equiv U_i|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$.

Prior to the formal analysis, we first idealize the communicated messages of our proposed protocol to alleviate the analysis between U_i and S_j, which are as follows:

Message 1: $U_i \rightarrow S_j : \langle P_{iJ}, CID_i, E_i, U_i \overset{h(SID_j\|h(y))}{\longleftrightarrow} S_j, D_i, N_i, M_1, M_2\rangle_{U_i \overset{A_i}{\longleftrightarrow} S_j}$.

Message 2: $S_j \rightarrow U_i : <SID_j, N_j, U_i \overset{A_i}{\longleftrightarrow} S_j>_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

Message 3: $U_i \rightarrow S_J : <SID_j, U_i \overset{A_i}{\longleftrightarrow} S_j, N_i, N_j>_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

Based on our proposed protocol, we make some initial state assumptions, which are listed as follows:

A_1: $U_i|\equiv \sharp(N_i)$;
A_2: $S_j|\equiv \sharp(N_j)$;
A_3: $U_i|\equiv (U_i \overset{A_i}{\longleftrightarrow} S_j)$;
A_4: $S_j|\equiv (U_i \overset{A_i}{\longleftrightarrow} S_j)$;
A_5: $U_i|\equiv S_j|\Rightarrow (U_i \overset{SKey_{ij}}{\longleftrightarrow} S_j)$;
A_6: $S_j|\equiv U_i|\Rightarrow (U_i \overset{SKey_{ij}}{\longleftrightarrow} S_j)$.

A_1 and A_2 believe that both U_i and S_j generate the fresh random numbers N_i and N_j respectively. Therefore, they assure their freshness, respectively. A_3 and A_4 are valid because the shared secret key A_i can be computed by both user U_i, and server S_j from the credentials issued by the *RC* as the master secret key. The assumption A_5 (A_6) holds because once U_i (S_j) shared the same shared session key $SKey_{ij}$.

Further, we demonstrate our proposed protocol based on the rules of the BAN logic that our protocol can achieve the intended goals using the initial assumptions, and the inside information descriptions are as follows:

According to the message 1, we could obtain:

Step 1: $S_j \triangleleft (CID_i, P_{ij}, U_i \overset{h(SID_j\|h(y))}{\longleftrightarrow} S_j, E_i, D_i, N_i, M_1, M_2)_{U_i \overset{A_i}{\longleftrightarrow} S_j}$.

From Step 1 and assumption A_3, we apply the message meaning rule to get:

Step 2: $S_j|\equiv U_i|\sim (U_i \overset{A_i}{\longleftrightarrow} S_j, N_i, U_i \overset{h(SID_j\|h(y)}{\longleftrightarrow} S_j)$.

From Step 2 and assumption A_1, we apply the freshness conjuncatenation rule to get:

Step 3: $S_j|\equiv \sharp(U_i \overset{A_i}{\longleftrightarrow} S_j, N_i, U_i \overset{h(SID_j\|h(y))}{\longleftrightarrow} S_j)$.

According to Steps 2 and 3, we apply the nonce-verification rule to obtain:

Step 4: $S_j|\equiv U_i|\equiv (U_i \overset{A_i}{\longleftrightarrow} S_j, N_i, U_i \overset{h(SID_j\|h(y))}{\longleftrightarrow} S_j)$.

According to Step 4, we apply the belief rule to obtain:

Step 5: $S_j|\equiv U_i|\equiv (U_i \overset{A_i}{\longleftrightarrow} S_j)$.

According to Step 5 and A_4, we apply the jurisdiction rule to get:

Step 6: $S_j|\equiv (U_i \overset{A_i}{\longleftrightarrow} S_j)$.

According to message 2, we obtain:

Step 7: $U_i \triangleleft (N_j, U_i \overset{A_i}{\longleftrightarrow} S_j, SID_j)_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

According to Step 7, we apply seeing rule and get:

Step 8: $U_i \triangleleft (M_3, M_4)$, where $M_4 = SK_{ij} \oplus N_j$ and $M_3 = h(SK_{ij}\|A_i\|SIDj\|N_j)$.

According to Step 8 and A_5, we apply the message meaning rule to get:

Step 9: $U_i|\equiv S_j|\sim (N_j, U_i \overset{A_i}{\longleftrightarrow} S_j, SID_j)_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

From Step 9 and assumption A_2, we apply the freshness conjuncatenation rule to get:

Step 10: $U_i|\equiv \sharp(N_j, U_i \overset{A_i}{\longleftrightarrow} S_j, SID_j)_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

According to Steps 9 and 10, we apply the nonce verification rule to obtain:

Step 11: $U_i|\equiv S_j|\equiv (N_j, U_i \overset{A_i}{\longleftrightarrow} S_j, SID_j)_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

According to Step 11, we apply the belief rule to get:

Step 12: $U_i|\equiv S_j|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$. (Goal 2)

According to A_5 and the Step 12, we apply the jurisdiction rule to obtain:

Step 13: $U_i|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$. (Goal 1)

According to message 3, we can obtain

Step 14: $S_j \triangleleft (N_i, U_i \overset{A_i}{\longleftrightarrow} S_j, N_j, SID_j)_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

According to Step 14 and assumption A_3, we apply the message meaning rule to get:

Step 15: $S_j|\equiv U_i|\sim (N_i, N_j, SID_j, U_i \overset{A_i}{\longleftrightarrow} S_j)_{U_i \overset{SK_{ij}}{\longleftrightarrow} S_j}$.

According to Step 15 and assumption A_2, we apply the freshness conjuncatenation rule to get:

Step 16: $S_j|\equiv \sharp(N_i, N_j, SID_j, U_i \overset{A_i}{\longleftrightarrow} S_j)$.

According to Step 16, we apply the belief rule to get:

Step 17: $S_j|\equiv U_i|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$. (Goal 4)

According to assumption A_4 and Step 17, we apply the jurisdiction rule to get:

Step 18: $S_j|\equiv (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j)$. (Goal 3)

According to Steps 12, 13, 17, and 18, it is clear that our protocol successfully achieves all the goals (Goals 1-4). Both user U_i and server S_j believe that they share a secure session key $h(SK_{ij}\|SID_j\|A_i\|N_i\|N_j\|D_i)$ with each other.

6.2 Discussion and Informal Security Analysis

In this section, we show that our scheme has the ability to support the important functionality properties and also to defend various known attacks.

6.2.1 Efficient Password Verification Mechanism

In our proposed scheme, during the login phase, the smart card can verify whether the user U_i inputs a correct password by checking if C_i^* is equal to C_i. If the user U_i inputs a wrong password $PW_i'(\neq PW_i)$, the smart card computes $b = L_i \oplus h(ID_i\|PW_i)$, $A_i = h(ID_i \oplus b \oplus PW_i')$ and $C_i^* = h(ID_i\|h(y)\|A_i)$ and gets $C_i^* \neq C_i$. Thus, the session is rejected by the smart

card. Therefore, the proposed scheme is efficient in the wrong password detection and improves the user friendliness.

6.2.2 User Anonymity

In our scheme, the anonymity of the user U_i is preserved by transmitting a session variant ID. The login request message $\{CID_i, P_{ij}, M_1, M_2\}$ sent to the server S_j, which does not contain ID in plain text. Each parameter in the login message is associated with nonce and secured through one-way hashing that provides dynamic to the message. This dynamic property helps in providing user anonymity.

6.2.3 Stolen Smartcard Attack

We assume that the user U_i's smart card has been lost or stolen. The attacker can extract the stored data $\{C_i, D_i, E_i, L_i, h(y), h(\cdot)\}$ from the smart card. Though the attacker can extract the stored information on the smart card, he/she cannot make use of the information. In order to login to the system, the attacker needs to know ID_i and PW_i correctly, which are not possible to guess both ID_i and password PW_i exactly at the same time as they are protected using a one-way cryptographic hash function. In addition, the master secret key x is unknown to the attacker. Therefore, the proposed scheme can resist stolen smart card attack.

6.2.4 Password Guessing Attack

In the login and authentication phases, the user U_i inputs ID_i and PW_i into the smart card. The computed values of b, A_i and C_i^* are securely protected using the one-way hash function and as discussed above the attacker needs to know ID_i and PW_i correctly, which are not possible. Therefore, our proposed scheme withstands password guessing attack.

6.2.5 Replay and Man-in-the-Middle Attacks

In our scheme, two random nonces N_i and N_j are generated by the user U_i and the server S_j, respectively, which make all messages dynamic and valid for that session only. An attacker can access the service by eavesdropping the previous login request $\{CID_i, P_{ij}, M_1, M_2\}$ from the user U_i, and may replay the same message to S_j. From the server S_j, the attacker gets an acknowledge message $\{M3, M4\}$. Despite of extracting all the transmitted messages, the attacker is not successful in computing the message $\{M_5\}$ to respond to the server S_j without knowing D_i, A_i and N_i. Similarly, if the attacker replies a previous message $\{M_3, M_4\}$ to U_i, due to the difference of the two random numbers N_i of these two different sessions, the previous computed random number N_j will not be equal to the random number N_j of this session that was chosen by S_j. Moreover, the computation of $h(D_i \| A_i \| N_j \| SID_j)$ will not be equal to M_3 due to which the authentication will fail. Therefore, our proposed scheme can resist replay and man-in-the-middle attack attack.

6.2.6 Forgery Attack

To forge as a legal user to login to the remote server S_j, an attacker must be able to eavesdrop a valid login request $\{CID_i, P_{ij}, M_1, M_2\}$ to fool S_j. However, the adversary cannot compute a valid login request message without knowing E_i, D_i, A_i and N_i. In

addition, if the adversary is a legal user of the system, he/she also cannot masquerade as another legal user to login to the remote server S_j as he/she cannot compute D_i and A_i from his/her smart card and the intercepted login request $\{CID_i, P_{ij}, M_1, M_2\}$ without knowing x, $h(x\|y)$, b and PW_i. Beside, if the adversary gets U_i's smart card and extracts the parameters $\{C_i, D_i, E_i, L_i, h(y), h(\cdot)\}$ stored in the smart card, he/she cannot also forge a login request to fool S_j, because he/she cannot use these parameters to compute the correct value of A_i without knowing the password PW_i. Therefore, our proposed scheme can withstand forgery attack.

6.2.7 Server and Registration Center Spoofing Attacks

An attacker being either legal but malicious user or legal but malicious server may try to perform the server spoofing attack and registration center spoofing attack in our proposed scheme. On the server spoofing attack, if the attacker is a legal user of the system, he/she must be able to forge a valid response message $\{M_3, M_4\}$ to U_i. However, the attacker cannot compute D_i and A_i from his/her smart card and the intercepted login request $\{CID_i, P_{ij}, M_1, M_2\}$ without the knowledge of $h(x\|y)$. Therefore, the attacker cannot compute the valid M_3 and M_4. Even if the attacker is a legal server of the system, he/she cannot also masquerade as another server to fool any legal user since he/she does not have the other server's secret information $h(SID_j\|h(y))$ to check the login request and cannot compute the valid response message $\{M_3, M_4\}$. Therefore, our proposed scheme can withstand the server spoofing attack.

On the registration center spoofing attack, the legal user and the legal server cannot get the master secret key x and secret number y, which are held by the registration center (RC) only. Therefore, any attacker cannot masquerade as the registration center, and our scheme can withstand this type of attack.

6.2.8 Known-Key Secrecy

Known-key secrecy means that compromise of one session key should not compromise other session keys. In our scheme, the session key $SKey_{ij} = h(SK_{ij}\|A_i\|SID_j\|N_i\|D_i\|N_j)$ between a user U_i and a server S_j is associated with D_i, A_i, N_i, N_j and SK_{ij}. If an attacker knows a past session key $SKey_{ij}$, he/she cannot obtain SK_{ij}, D_i and A_i from the session key since they are protected by the one-way hash function $h(\cdot)$. As a result, the attacker cannot get the other session keys. Thus, our scheme provides the known-key secrecy property.

6.2.9 Forward Secrecy

Forward secrecy means that if the master secret key x of the system is compromised, the secrecy of previously established session keys should not be affected. If the master secret key x is compromised for some reason, the attacker cannot compute any previous session key $SKey_{ij}$ between a user U_i and a server S_j without knowing PW_i, b, ID_i and y. Therefore, the proposed scheme can ensure forward secrecy property.

6.2.10 Denial-of-Service Attack

In our proposed scheme, U_i sends the login message $\{CID_i, P_{ij}, M_1, M_2\}$ to the opted server S_j to get the services. Upon receiving the login message, S_j performs some

computations and also verifies the legitimacy of U_i. Here, in our proposed scheme, S_j performs the computations as shown in Step $V1$ and verifies the login request as shown in Step $V2$ in Sect. 5.3. If the verification fails, S_j terminates the session. But, once the legitimacy of the user is verified, S_j agrees to provide the necessary services to U_i. Therefore, our proposed scheme is secure against this attack.

6.2.11 Good Repairability

A good login and authentication mechanism should be able to revoke lost or stolen smart card with good repairability. The registration center RC should be able to revoke the smart card, if at all it is lost or stolen. In our scheme, if a legitimate user U_i's smart card is lost or stolen, he/she has to request the RC to issue the new smart card with the same user identity ID_i for his/her future communications. In order to launch the login request with the help of stolen smart card, U_i needs to have valid login credentials ID_i and PW_i. We have already shown that our scheme preserves user anonymity, resists stolen smart card attack and off-line password guessing attack. Therefore, it is not possible to get valid ID_i and PW_i. Hence, our scheme provides good repairability property.

6.2.12 Proper Mutual Authentication

The proposed scheme provides proper mutual authentication due to the following reason. The user U_i sends the message $\{CID_i, P_{ij}, M_1, M_2\}$ to the server S_j to access the service. After receiving the message, S_j computes N_i, E_i, B_i, D_i, A_i and then checks if $h(P_{ij}\|CID_i\|A_i\|N_i) = M_1$. If it holds, U_i is a valid user and the login request is accepted by the server S_j. Otherwise, S_j rejects the login request. Since the authentication equation relies on the one-way hash function, any fabricated message $\{CID'_i, P'_{ij}, M'_1, M'_2\}$ cannot pass the verification. Then, S_j computes the message $\{M_3, M_4\}$ and sends it to U_i. U_i computes $SK_{ij} = h(D_i\|A_i)$ and N_j, and checks whether $h(SK_{ij}\|A_i\|SID_j\|N_j)$ is equal to M_3. If they are not equal, U_i terminates the scheme. Otherwise, S_j is authenticated by U_i. Since it protected by a one-way hash function, any fabricated message $\{M'_3, M'_4\}$ cannot pass the authentication. With the same reason, any fabricated mutual authentication message $\{M'_5\}$ cannot pass the mutual authentication. Therefore, our proposed scheme provides proper mutual authentication.

7 Simulation for Formal Security Verification using AVISPA Tool

In this section, we simulate our scheme for the formal security verification using the widely-accepted AVISPA (Automated Validation of Internet Security Protocols and Applications) tool.

AVISPA is a modular and expressive formal language for specifying protocols and their security properties. It integrates different back-ends that implement a variety of state-of-the-art automatic analysis techniques [1]. It is a push-button tool for the automated validation of Internet security-sensitive protocols and applications, which becomes a widely-accepted tool for our formal security verification in recent years, [4, 6, 7, 28–35]. AVISPA contains four back-ends: On-the-fly Model-Checker (OFMC), Constraint Logic based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC) and Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP). The detailed descriptions of these back-ends are available in [1].

The protocols to be analyzed under the AVISPA tool need to be specified in the HLPSL (High Level Protocols Specification Language), which is a role-oriented language [1]. 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. In HLPSL, an intruder is always denoted by *i*, which is always modeled using the Dolev–Yao model [9]. Thus, it is possible for the intruder to assume a legitimate role in a protocol run. The role system also defines a number of sessions, and a number of principals and some basic roles. HLPSL is first translated using HLPSL2IF translator to the intermediate format (IF). IF is then fed to one of the backends to produce the output format (OF). It contains the following sections [45]:

- SUMMARY section tells that whether the tested protocol is safe, unsafe, or whether the analysis is inconclusive.
- DETAILS section either explains under what condition the tested protocol is declared safe, or what conditions have been used for finding an attack, or finally why the analysis was inconclusive.
- PROTOCOL, GOAL and BACKEND sections denote the name of the protocol, the goal of the analysis and the name of the back-end used, respectively.
- Finally, after some comments and statistics, the trace of an attack (if any) is also printed in the standard Alice-Bob format.

Several basic types are supported in HLPSL, which are given below for better understanding of the specifications of various roles described in Sect. 7.1 [1]:

- *agent:* It is for the principal name. The intruder is always assumed to have the special identifier *i*.
- *public_key:* It indicates agents' public keys in a public-key cryptosystem. For example, given a public (respectively private) key *K*, its inverse private (respectively public) key is obtained by *inv_K*.
- *symmetric_key:* It represents the keys for a symmetric-key cryptosystem.
- *text:* It is often used as nonces. These values can be also used for messages. For example, if *Ni* is of type *text (fresh)*, then Ni' will be a fresh value which the intruder cannot guess easily.
- *nat:* It represents the natural numbers in non-message contexts.
- *const:* It denotes the constants.
- *hash_func:* It represents cryptographic hash functions.

For a given message *msg* and encryption key *k*, $\{msg\}\_k$ denotes the symmetric/public-key encryption. In HLPSL, the associative "$\cdot$" operator is always used for concatenation.

7.1 Specifying the Protocol

This section describes the specifications of the roles in HLPSL for our scheme. Three basic roles for a user U_i, the *RC* and a server S_j are implemented in HLPSL for our scheme. Apart from these roles, the roles for the session, goal and environment in HLPSL must be specified for our scheme. We have implemented our scheme for the formal security verification during the registration phase, login phase as well as authentication and session key agreement phase.

The role of the initiator, user U_i is shown in Fig. 2. U_i first receives the start signal, changes its state from 0 to 1 maintained by the variable *State*, and then sends the

Fig. 2 Role specification in HLPSL for a user U_i

```
role user(Ui, RC, Sj : agent,
% symmetric key between Ui and RC
      SKuirc : symmetric_key,
% H is hash function
       H : hash_func,
       SEND, RECV: channel(dy))
played_by Ui
def=
 local State : nat,
    IDi, SIDj, PWi, B, X, Y : text,
    Ai, Bi, Ci, Di, Ei, Ni, Nj : text,
    CIDi, Pij, M1, M2, M3, M4 : text,
    SKij, M5 : text
 const user_server_ni, server_user_nj,
     sub1, sub2, sub3 : protocol_id
init State := 0
 transition
% User registration phase
% Ui sends < IDi, Ai > to RC via a secure channel
1. State = 0 /\ RECV(start) =|>
State' := 2  /\ SEND({IDi.H(xor(IDi, xor(B,PWi)))}_SKuirc)
          /\ secret({PWi,B}, sub1, Ui)
          /\ secret({IDi,SIDj}, sub2, {Ui,RC,Sj})
% Ui receives <smart card> from RC via a secure channel
2. State = 2 /\ RECV({H(IDi.H(Y).H(xor(IDi, xor(B,PWi)))).
              H(H(H(xor(IDi, xor(B,PWi))).X).H(X.Y)).
              xor(H(H(xor(IDi, xor(B,PWi))).X),H(X.Y)).
              H(Y).H}_SKuirc) =|>
% Login phase
% Ui sends < CIDi, Pij, M1, M2 > to Sj via a public channel
State' := 4  /\ secret({X,Y}, sub3, RC)
          /\ Ni' := new()
          /\ CIDi' := xor(H(xor(IDi, xor(B,PWi))),
             H(H(H(H(xor(IDi, xor(B,PWi))).X).
             H(X.Y)).SIDj.Ni'))
          /\ Pij' := xor(xor(H(H(xor(IDi, xor(B,PWi))).X),
                H(X.Y)), H(H(SIDj.H(Y)).Ni'))
          /\ M1' := H(Pij'.CIDi'.H(xor(IDi, xor(B,PWi))).Ni')
          /\ M2' := xor(H(SIDj.H(Y)), Ni')
          /\ SEND(CIDi'.Pij'.M1'. M2')
% Ui has freshly generated the value Ni' for Sj
         /\ witness(Ui, Sj, user_server_ni, Ni')
% Verification phase
% Ui receives < M3, M4 > from Sj via a public channel
3. State = 4 /\ RECV(H(H(H(H(xor(IDi, xor(B,PWi))).X).
               H(X.Y)).H(xor(IDi, xor(B,PWi)))).
               H(xor(IDi, xor(B,PWi))).SIDj.Nj').
               xor(H(H(H(H(xor(IDi, xor(B,PWi))).X).
              H(X.Y)).H(xor(IDi, xor(B,PWi)))),Nj')) =|>
% Ui sends < M5 > to Sj via a public channel
State' := 6  /\ M5' := H(H(H(H(xor(IDi, xor(B,PWi))).X).
                 H(X.Y)).H(xor(IDi, xor(B,PWi)))).
                 H(xor(IDi, xor(B,PWi))).SIDj.Ni.Nj')
          /\ SEND(M5')
% Ui's acceptance of the value Nj generated for Ui by Sj
          /\ request(Sj, Ui, server_user_nj, Nj')
end role
```

registration request message $\{ID_i, A_i\}$ securely to the *RC* during the registration phase with the help of *SEND*( ) operation. U_i also receives a smart card with the information $\{C_i, D_i, E_i, h(y), h(\cdot)\}$ securely from the *RC* with the help of *RECV*( ) operation. During the login, and authentication and key agreement phases, U_i sends the login request message $\{CID_i, P_{ij}, M_1, M_2\}$ to S_j via a public channel. U_i then receives the message $\{M_3, M_4\}$ from S_j via a public channel. Finally, U_i sends the message $\{M_5\}$ to S_j via a public channel.

The type declaration *channel* (dy) declares that the channel is for the Dolev–Yao threat model [9], which means that the intruder (i) has the ability to intercept, analyze, and/or modify messages transmitted over a insecure public channel.The "*played_by A*" declaration indicates that the agent named in variable *A* plays in the role. A knowledge declaration (generally in the top-level *Environment* role) is used to specify the intruder's initial knowledge. Immediate reaction transitions are of the form $X = | > Y$, which relate an event *X* and an action *Y*. Note that the declaration witness(A, B, id, E) declares for a (weak) authentication property of *A* by *B* on *E*, declares that agent *A* is witness for the information *E*; this goal will be identified by the constant *id* in the goal section [1]. The declaration request (B, A, id, E) hints for a strong authentication property of *A* by *B* on *E*, declares that agent *B* requests a check of the value *E*; this goal will be identified by the constant *id* in the goal section [1]. For examples, witness(Ui, Sj, user_server_ni, Ni') means that U_i has freshly generated the value N_i for S_j. request(Sj, Ui, server_user_nj, Nj') tells that U_i's acceptance of the value N_j generated for U_i by S_j. The declaration type secret(PWi,B, sub1, Ui) means that the information PW_i and b are kept secret to the user U_i only, which are characterized by the protocol id *sub*1. If a variable *V* is kept permanently secret, it is expressed by the goal *secrecy_of V*. As a result, if *V* is ever obtained or derived by the intruder, a security violation will result. In a similar way, the roles of the *RC* and a server S_j are given in Figs. 3 and 4, respectively.

The roles for the session, and the goal and environment of our scheme are given in Fig. 5. All the basic roles, such as user, rc and server are the instances with concrete arguments in the role of the session. The top-level role (environment) is always defined in HLPSL specification. The intruder (i) also participates in the execution of protocol as a concrete session as shown in Fig. 5. We have three secrecy goals and two authentication goals in our implementation. For example, the secrecy goal: secrecy_of sub1 tells that the information PW_i and b are kept secret to the user U_i only. The authentication goal: authentication_on user_server_ni indicates that U_i generates a random nonce N_i, where N_i is only known to U_i. When the server S_j will receive N_i from other messages from U_i, it performs a strong authentication for U_i based on N_i.

7.2 Simulation Results

We have simulated our scheme under the OFMC and CL-AtSe backends using the Security Protocol ANimator for AVISPA (SPAN) [1]. The following verifications are executed in our scheme [6, 7, 28–32]:

– *Executability check on non-trivial HLPSL specifications:* Due to some modeling mistakes, the protocol model can not sometimes execute to completion. It may be possible that the AVISPA backends can not find an attack, if the protocol model can not reach to a state where that attack can happen. An executability test becomes extremely essential [45]. Our scheme shows that the protocol description is well matched with the designed goals as specified in Figs. 2, 3, 4 and 5 for the executability test.

Fig. 3 Role specification in HLPSL for the *RC*

```
role rc (Ui, RC, Sj: agent,
% symmetric key between Ui and RC
      SKuirc : symmetric_key,
% H is hash function
       H : hash_func,
       SEND, RECV: channel(dy))
played_by RC
def=
 local State : nat,
    IDi, SIDj, PWi, B, X, Y : text,
    Ai, Bi, Ci, Di, Ei, Ni, Nj : text,
    CIDi, Pij, M1, M2, M3, M4 : text,
    SKij, M5 : text
 const user_server_ni, server_user_nj,
     sub1, sub2, sub3 : protocol_id
init State := 0
 transition
% User registration phase
% RC receives <IDi, Ai> from Ui via a secure channel
1. State = 0 /\ RECV({IDi.H(xor(IDi, xor(B,PWi)))}_SKuirc) =|>
State' := 1  /\ secret({PWi,B}, sub1, Ui)
         /\ secret({IDi,SIDj}, sub2, {Ui,RC,Sj})
% RC sends < smart card > to Ui via a secure channel
         /\ Ai' := H(xor(IDi, xor(B,PWi)))
         /\ Bi' := H(Ai'.X)
         /\ Ci' := H(IDi.H(Y).Ai')
         /\ Di' := H(Bi'.H(X.Y))
         /\ Ei' := xor(Bi',H(X.Y))
         /\ SEND({Ci'.Di'.Ei'.H(Y).H}_SKuirc)
         /\ secret({X,Y}, sub3, RC)
end role
```

- *Replay attack check:* For the replay attack check, the OFMC and Cl-AtSe back-ends verify if the legitimate agents can execute the specified protocol by performing a search of a passive intruder. These back-ends provide the intruder the knowledge of some normal sessions between the legitimate agents. The test results shown in Figs. 6 and 7 indicate that our scheme is secure against the replay attack.
- *Dolev–Yao model check:* For the Dolev–Yao model check, the OFMC and Cl-AtSe back-ends also verifies whether there is any man-in-the-middle attack possible by an intruder. It is evident from the results reported in Figs. 6 and 7 that our scheme fulfills the design properties and is also secure under these backends.

8 Performance Comparison with Related Schemes

In this section, we compare the performance and functionality features of our proposed scheme with the related existing authentication schemes proposed for the multi-server environment. This evaluation gives an insight into the effectiveness of the proposed scheme.

In Table 5, we have shown the security features provided and protected by our scheme as compared to those for the existing related schemes, such as Hsiang-Shih's scheme [12], Lee et al.'s scheme [19], Li et al.'s scheme [21] and Shunmuganathan et al.'s scheme [38]. It is noted that password guessing attack is possible all other existing schemes. Moreover, proper mutual authentication, good repairability and two-factor

Fig. 4 Role specification in HLPSL for the server S_j

```
role server (Ui, RC, Sj : agent,
% symmetric key between Ui and RC
      SKuirc : symmetric_key,
% H is hash function
       H : hash_func,
       SEND, RECV: channel(dy))
played_by Sj
def=
 local State : nat,
    IDi, SIDj, PWi, B, X, Y : text,
    Ai, Bi, Ci, Di, Ei, Ni, Nj : text,
    CIDi, Pij, M1, M2, M3, M4 : text,
    SKij, M5 : text
 const user_server_ni, server_user_nj,
      sub1, sub2, sub3 : protocol_id
init State := 0
transition
% Logic phase
% Sj receives < CIDi, Pij, M1, M2 > from Ui via a public channel
1. State = 0 /\ RECV(xor(H(xor(IDi, xor(B,PWi))),
            H(H(H(H(xor(IDi, xor(B,PWi))).X).
            H(X.Y)).SIDj.Ni')).
            xor(xor(H(H(xor(IDi, xor(B,PWi))).X),
            H(X.Y)), H(H(SIDj.H(Y)).Ni')).
            H(Pij'.CIDi'.H(xor(IDi, xor(B,PWi))).Ni').
            xor(H(SIDj.H(Y)), Ni')) =|>
State' := 3  /\ secret({PWi,B}, sub1, Ui)
         /\ secret({IDi,SIDj}, sub2, {Ui,RC,Sj})
         /\ secret({X,Y}, sub3, RC)
% Verification phase
         /\ Nj' := new()
         /\ SKij' := H(H(H(H(xor(IDi, xor(B,PWi))).X).
                H(X.Y)).H(xor(IDi, xor(B,PWi))))
         /\ M3' := H(SKij'.H(xor(IDi, xor(B,PWi))).SIDj.Nj')
         /\ M4' := xor(SKij',Nj')
% Sj sends < M3, M4 > to Ui via a public channel
         /\ SEND(M3'.M4')
% Sj has freshly generated the value Nj' for Ui
        /\ witness(Sj, Ui, server_user_nj, Nj')
% Sj receives < M5 > from Ui via a public channel
2. State = 3 /\ RECV(H(H(H(H(xor(IDi, xor(B,PWi))).X).
              H(X.Y)).H(xor(IDi, xor(B,PWi)))).
              H(xor(IDi, xor(B,PWi))).SIDj.Ni'.Nj')) =|>
% Sj's acceptance of the value Ni generated for Sj by Ui
State' := 5 /\ request(Ui, Sj, user_server_ni, Ni')
end role
```

security are not provided in the existing schemes. In addition, existing schemes are vulnerable to impersonation attack and forgery attack. We also see that the stolen smart card attack is not protected by the existing schemes [19, 21, 38]. On the other hand, our scheme protects known attacks shown in this table, and also supports good features mentioned in the table.

In Table 6, we have compared the communication overhead of our scheme with the existing related schemes, such as Hsiang-Shih's scheme [12], Lee et al.'s scheme [19], Li et al.'s scheme [21] and Shunmuganathan et al.'s scheme [38] during the login and authentication phases in terms of the number of transmitted messages and the number of bits required for these messages. For the communication cost analysis, we assume that each of the identity ID_i of the user U_i and the identity SID_j of the serverS_j is 160 bits. The

Fig. 5 Role specification in HLPSL for the session, and goal and environment

```
role session(Ui, RC, Sj : agent,
% symmetric key between Ui and RC
      SKuirc : symmetric_key,
% H is hash function
       H : hash_func)
def=
 local SN1, SN2, SN3, RV1, RV2, RV3 : channel (dy)
composition
  user(Ui, RC, Sj, SKuirc, H, SN1, RV1)
/\ rc(Ui, RC, Sj, SKuirc, H, SN2, RV2)
/\ server(Ui, RC, Sj, SKuirc, H, SN3, RV3)
end role

role environment()
def=
 const ui, rc, sj: agent,
       skuirc : symmetric_key,
       h : hash_func,
       user_server_ni, server_user_nj,
       sub1, sub2, sub3 : protocol_id
intruder_knowledge = {ui, rc, sj, h}
composition
  session(ui, rc, sj, skuirc, h)
/\ session(i, rc, sj, skuirc, h)
/\ session(ui, i, sj, skuirc, h)
/\ session(ui, rc, i, skuirc, h)
end role
goal
 secrecy_of sub1
 secrecy_of sub2
 secrecy_of sub3
 authentication_on user_server_ni
 authentication_on server_user_nj
end goal
environment()
```

random nonce is assumed to be 160 bits. The hash output or message digest is taken as 160 bits (if we use SHA-1 as the secure one-way hash function [37]). In our scheme, we need three messages to be transmitted between U_i and S_j. During the login phase, the message $\{CID_i, P_{ij}, M_1, M_2\}$ requires $4 \times 160 = 640$ bits. During the authentication and key agreement phase, the messages $\{M_3, M_4\}$ and $\{M_5\}$ require $2 \times 160 = 320$ bits and 160 bits, respectively. As a result, total communication overhead of our scheme during the login and authentication phases becomes $(640 + 320 + 160) = 1120$ bits for three transmitted messages. On the other hand, the communication overheads for Hsiang-Shih's scheme, Lee et al.'s scheme, Li et al.'s scheme and Shunmuganathan et al's scheme are 2720, 1280, 1120 and 1120 bits, respectively. Note that our scheme performs better than Hsiang-Shih's scheme and Lee et al.'s scheme, whereas the communication costs required for Li et al.'s scheme and Shunmuganathan et al.'s scheme are same as that for our scheme. However, our scheme is more secure and provides more functionality features as compared to other schemes as shown in Table 5.

Fig. 6 The result of the analysis using OFMC backend

```
% OFMC
% Version of 2006/02/13
SUMMARY
 SAFE
DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
 C:\progra~1\SPAN\testsuite\results
 \authentication.if
GOAL
 as_specified
BACKEND
 OFMC
COMMENTS
STATISTICS
 parseTime: 0.00s
 searchTime: 77.92s
 visitedNodes: 3191 nodes
 depth: 9 plies
```

Fig. 7 The result of the analysis using CL-AtSe backend

```
SUMMARY
 SAFE

DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
 TYPED_MODEL

PROTOCOL
 C:\progra~1\SPAN\testsuite\results
 \authentication.if

GOAL
 As Specified

BACKEND
 CL−AtSe

STATISTICS
 Analysed  : 19 states
 Reachable  : 4 states
 Translation: 0.08 seconds
 Computation: 3.05 seconds
```

Finally, in Table 7, we have compared the computation overhead of our scheme with the existing Hsiang-Shih's scheme, Lee et al.'s scheme, Li et al.'s scheme and Shunmuganathan et al's scheme. Let T_h and T_x refer to the execution time of the one way hash function operation and bitwise XOR operation, respectively. In our scheme, during the login phase, the computational cost is $8T_h + 6T_x$. During the authentication and key agreement phase, our scheme requires the computation cost $17T_h + 6T_x$. As a result, total computation overhead becomes $25T_h + 12T_x$. On the other hand, the computation overheads for Hsiang-Shih's scheme, Lee et al.'s scheme, Li et al.'s scheme and Shunmuganathan et al's scheme are $26T_h + 23T_x$, $20T_h + 10T_x$, $19T_h + 12T_x$ and $20T_h + 10T_x$, respectively. Due to efficient one-way hash function and bitwise XOR operations, the

Table 5 Comparison of security features

Security attributes	[12]	[19]	[21]	[38]	Ours
S_1	×	×	×	×	√
S_2	√	√	√	√	√
S_3	√	√	√	√	√
S_4	√	×	×	×	√
S_5	×	×	×	×	√
S_6	√	√	√	√	√
S_7	×	×	×	×	√
S_8	×	×	×	×	√
S_9	×	×	×	×	√
S_{10}	×	×	×	×	√
S_{11}	√	√	√	√	√
S_{12}	√	×	√	√	√
S_{13}	√	√	√	√	√

S_1: password guessing attack; S_2: privileged-insider attack; S_3: user anonymity; S_4: stolen smart card attack; S_5: impersonation attack; S_6: replay attack; S_7: proper mutual authentication; S_8: good reparability; S_9: forgery attack; S_{10}: two-factor security; S_{11}: session key agreement; S_{12}: efficient password change; S_{13}: without verification table; √ = preserved; × = not preserved

Table 6 Communication overhead comparison during the login and authentication phases

Scheme	Total number of messages	Total number of bits
Hsiang-Shih [12]	5	2720
Lee et al. [19]	3	1280
Li et al. [21]	3	1120
Shunmuganathan et al. [38]	3	1120
Ours	3	1120

Table 7 Computation overhead comparison during the login and authentication phases

Scheme	Login phase	Authentication phase	Total cost
Hsiang et al. [12]	$9T_h + 9T_x$	$17T_h + 14T_x$	$26T_h + 23T_x$
Lee et al. [19]	$5T_h + 6T_x$	$15T_h + 4T_x$	$20T_h + 10T_x$
Li et al. [21]	$6T_h + 4T_x$	$13T_h + 8T_x$	$19T_h + 12T_x$
Shunmuganathan et al. [38]	$7T_h + 3T_x$	$13T_h + 7T_x$	$20T_h + 10T_x$
Ours	$8T_h + 6T_x$	$17T_h + 6T_x$	$25T_h + 12T_x$

computation overhead of our scheme is also comparable with that for Hsiang-Shih's scheme, Lee et al.'s scheme, Li et al.'s scheme and Shunmuganathan et al's scheme.

9 Conclusion

In this paper, we have first reviewed Shunmuganathan et al.'s remote user authentication scheme for multi-server environment, and then shown that Shunmuganathan et al.'s scheme is vulnerable to password guessing attack, user impersonation attack, stolen smart card's attack, replay attack, forgery attack. In addition, Shunmuganathan et al.'s

scheme fails to provide forward secrecy and two-factor security. To overcome the shortcomings of these security weaknesses, we have proposed a more secure and effective multi-server authentication scheme based on the dynamic ID. We have demonstrated that our scheme can resist and sustain to the essential requirements for multi-server environment. In comparison with Shunmuganathan et al.'s scheme and other related schemes, our proposed scheme is more secure. Furthermore, we have simulated our scheme for the formal security verification using the most-widely accepted and used AVISPA tool and the simulation results clearly show that our scheme is secure. Through the widely-accepted BAN logic, we have also proved that our scheme provides mutual authentication between a user and a server. In addition, our scheme is comparable in both communication and computation as compared to Shunmuganathan et al.'s scheme and other schemes. High security along with low communication and computation overheads make our proposed scheme more suitable for practical applications in wireless communications.

Acknowledgements The authors would like to acknowledge the many helpful suggestions of the anonymous reviewers and the Editor of this Journal, which have improve the content and presentation of the paper.

References

1. AVISPA. Automated Validation of Internet Security Protocols and Applications. http://www.avispa-project.org/. Accessed on January 2015.
2. Burrows, M., Abadi, M., & Needham, R. (1990). A logic of authentication. *ACM Transactions on Computer Systems*, *8*(1), 18–36.
3. Chang, C. C., & Lee, J. S. (2004). An efficient and secure multi-server password authentication scheme using smart cards. In *International conference on cyberworlds (Cw 2004)* (pp. 417–422) Tokyo.
4. Chatterjee, S., Roy, S., Das, A. K., Chattopadhyay, S., Kumar, N., & Vasilakos, A. V. (2016). Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment. *IEEE Transactions on Dependable and Secure Computing*,. doi:10.1109/TDSC.2016.2616876.
5. Damgård, I. B. (1990). A design principle for hash functions. In *9th Annual international cryptology conference (CRYPTO'89)* (pp. 416–427) Santa Barbara.
6. Das, A. K. (2015). A secure and efficient user anonymity-preserving three-factor authentication protocol for large-scale distributed wireless sensor networks. *Wireless Personal Communications*, *82*(3), 1377–1404.
7. Das, A. K. (2016). A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks. *Peer-to-Peer Networking and Applications*, *9*(1), 223–244.
8. Das, M. L., Saxena, A., & Gulati, V. P. (2004). A dynamic id-based remote user authentication scheme. *IEEE Transactions on Consumer Electronics*, *50*(2), 629–631.
9. Dolev, D., & Yao, C. A. (1983). On the security of public key protocols. *IEEE Transactions on Information Theory*, *29*(2), 198–208.
10. Eisenbarth, T., Kasper, T., Moradi, A., Paar, C., Salmasizadeh, M., & Shalmani, M., et al. (2008). On the power of power analysis in the real world: A complete break of the keeloq code hopping scheme. In *28th Annual international cryptology conference (CRYPTO 2008)*, volume 5157 of lecture notes in computer science California, USA (pp. 203–220) Santa Barbara.
11. He, D., Chen, J., & Hu, J. (2012). An ID-based client authentication with key agreement protocol for mobile clientserver environment on ECC with provable security. *Information Fusion*, *13*(3), 223–230.
12. Hsiang, H. C., & Shih, W. K. (2009). Improvement of the secure dynamic id based remote user authentication scheme for multi-server environment. *Computer Standards and Interfaces*, *31*(6), 1118–1123.
13. Hwang, M. S., & Li, L. H. (2000). A new remote user authentication scheme using smart cards. *IEEE Transactions on Consumer Electronics*, *46*(1), 28–30.
14. Juang, W. S. (2004). Efficient multi-server password authenticated key agreement using smart cards. *IEEE Transactions on Consumer Electronics*, *50*(1), 251–255.
15. Juang, W. S., Chen, S. T., & Liaw, H. T. (2008). Robust and efficient password-authenticated key agreement using smart cards. *IEEE Transactions on Industrial Electronics*, *55*(6), 2551–2556.

16. Kocher, P., Jaffe, J., & Jun, B. (1999). Differential power analysis. In *19th Annual international cryptology conference (CRYPTO'99)*. LNCS California (Vol. 1666, pp. 388–397) Santa Barbara.
17. Ku, W. C., & Chang, S. T. (2005). Impersonation attack on a dynamic id-based remote user authentication scheme using smart cards. *IEICE Transactions on Communications*, *E88–B*(5), 2165–2167.
18. Lamport, L. (1981). Password authentication with insecure communication. *Communications of the ACM*, *24*(11), 770–772.
19. Lee, C. C., Lai, Y. M., & Li, C. T. (2012). An improved secure dynamic id based remote user authentication scheme for multi-server environment. *International Journal of Security and Its Applications*, *6*(2), 203–209.
20. Lee, C. C., Lin, T. H., & Chang, R. X. (2011). A secure dynamic id based remote user authentication scheme for multi-server environment using smart cards. *Expert Systems with Applications*, *38*(11), 13863–13870.
21. Li, X., Ma, J., Wang, W., Xiong, Y., & Zhang, J. (2013). A novel smart card and dynamic id based remote user authentication scheme for multi-server environments. *Mathematical and Computer Modelling*, *58*(1), 85–95.
22. Li, X., Niu, J., Kumari, S., Liao, J., & Liang, W. (2015). An enhancement of a smart card authentication scheme for multi-server architecture. *Wireless Personal Communications*, *80*(1), 175–192.
23. Li, X., Xiong, Y., Ma, J., & Wang, W. (2012). An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards. *Journal of Network and Computer Applications*, *35*(2), 763–769.
24. Liao, Y. P., & Wang, S. S. (2009). A secure dynamic id based remote user authentication scheme for multi-server environment. *Computer Standards and Interfaces*, *31*(1), 24–29.
25. Lin, I. C., Hwang, M. S., & Li, L. H. (2003). A new remote user authentication scheme for multi-server architecture. *Future Generation Computer Systems*, *19*(1), 13–22.
26. Messerges, T. S., Dabbish, E. A., & Sloan, R. H. (2002). Examining smart-card security under the threat of power analysis attacks. *IEEE Transactions on Computers*, *51*(5), 541–552.
27. Mishra, D., Das, A. K., & Mukhopadhyay, S. (2014). A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards. *Expert Systems with Applications*, *41*(18), 8129–8143.
28. Mishra, D., Das, A. K., & Mukhopadhyay, S. (2016). A secure and efficient ECC-based user anonymity-preserving session initiation authentication protocol using smart card. *Peer-to-Peer Networking and Applications*, *9*(1), 171–192.
29. Odelu, V., Das, A. K., & Goswami, A. (2015). DMAMA: Dynamic migration access control mechanism for mobile agents in distributed networks. *Wireless Personal Communications*, *84*(1), 207–230.
30. Odelu, V., Das, A. K., & Goswami, A. (2015). An effective and robust secure remote user authenticated key agreement scheme using smart cards in wireless communication systems. *Wireless Personal Communications*, *84*(4), 2571–2598.
31. Odelu, V., Das, A. K., & Goswami, A. (2015). A secure and scalable group access control scheme for wireless sensor networks. *Wireless Personal Communications*, *85*(4), 1765–1788.
32. Odelu, V., Das, A. K., & Goswami, A. (2015). A secure biometrics-based multi-server authentication protocol using smart cards. *IEEE Transactions on Information Forensics and Security*, *10*(9), 1953–1966.
33. Odelu, V., Das, A. K., & Goswami, A. (2016). SEAP: Secure and efficient authentication protocol for NFC applications using pseudonyms. *IEEE Transactions on Consumer Electronics*, *62*(1), 30–38.
34. Odelu, V., Das, A. K., Wazid, M., & Conti, M. (2016). Provably secure authenticated key agreement scheme for smart grid. *IEEE Transactions on Smart Grid*,. doi:10.1109/TSG.2016.2602282.
35. Reddy, A. G., Das, A. K., Yoon, E.-J., & Yoo, K.-Y. (2016). A secure anonymous authentication protocol for mobile services on elliptic curve cryptography. *IEEE Access*, *4*, 4394–4407.
36. Sarkar, P. (2010). A simple and generic construction of authenticated encryption with associated data. *ACM Transactions on Information and System Security*, *13*(4), 33.
37. Secure hash standard. (1995). FIPS PUB 180-1, National Institute of Standards and Technology (NIST), U.S. Department of Commerce, April 1995. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2. Accessed on January 2015.
38. Shunmuganathan, S., Saravanan, R. D., & Palanichamy, Y. (2015). Secure and efficient smart-card-based remote user authentication scheme for multiserver environment. *Canadian Journal of Electrical and Computer Engineering*, *38*(1), 20–30.
39. Sood, S. K., Sarje, A. K., & Singh, K. (2011). A secure dynamic identity based authentication protocol for multi-server architecture. *Journal of Network and Computer Applications*, *34*(2), 609–618.
40. Stallings, W. (2006). *Cryptography and network security: Principles and practice, 5/E*. Upper Saddle River, NJ: Prentice Hall Press.

41. Stinson, D. R. (2006). Some observations on the theory of cryptographic hash functions. *Designs, Codes and Cryptography*, *38*(2), 259–277.
42. Sun, H. M. (2000). An efficient remote use authentication scheme using smart cards. *IEEE Transactions on Consumer Electronics*, *46*(4), 958–961.
43. Tsai, J. L. (2008). Efficient multi-server authentication scheme based on one-way hash function without verification table. *Computers and Security*, *27*(3), 115–121.
44. Tsaur, W. J., Wu, C. C., & Lee, W. B. (2004). A smart card-based remote scheme for password authentication in multi-server internet services. *Computer Standards and Interfaces*, *27*(1), 39–51.
45. von Oheimb, D. (2005). The high-level protocol specification language hlpsl developed in the eu project avispa. In *Proceedings of APPSEM 2005 workshop* (pp. 1–17) Frauenchiemsee.
46. Wang, D., He, D., Wang, P., & Chu, C.-H. (2015). Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment. *IEEE Transactions on Dependable and Secure Computing*, *12*(4), 428–442.
47. Xue, K., Hong, P., & Ma, C. (2014). A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture. *Journal of Computer and System Sciences*, *80*(1), 195–206.
48. Yang, W. H., & Shieh, S. P. (1999). Password authentication schemes with smart cards. *Computers and Security*, *18*(8), 727–733.
49. Zhao, D., Peng, H., Li, S., & Yang, Y. (2013). An efficient dynamic id based remote user authentication scheme using self-certified public keys for multi-server environment. arXiv:1305.6350.

Srinivas Jangirala completed his Bachelor of Science in 2003 from Kakatiya University, India. He has received Master of Science degree from Kakatiya University in 2008. He has received Master of Technology degree from IIT Kharagpur in 2011. Currently, he is pursuing his Ph.D from the Department of Mathematics, IIT Kharagpur. His research interests include authentication protocols, information security, digital rights management, cloud computing.

Sourav Mukhopadhyay completed his B.Sc (Honours in Mathematics) in 1997 from University of Calcutta, India. He has done M.Stat (in statistics) and M.Tech (in computer science) from Indian Statistical Institute, India, in 1999 and 2001 respectively. He received his Ph.D. degree in the area of Cryptology (Computer Science) from Indian Statistical Institute, India in 2007. Currently, he is an Associate Professor in the Department of Mathematics, IIT Kharagpur. His research and teaching interests include network security, cryptology, mathematics, statistics and computer science. He has published more than 60 papers in international journals and conferences in these areas.

Ashok Kumar Das received the Ph.D. degree in computer science and engineering, the M.Tech. degree in computer science and data processing, and the M.Sc. degree in mathematics from IIT Kharagpur, India. He is currently an Assistant Professor with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, India. His current research interests include cryptography, wireless sensor network security, hierarchical access control, data mining, security in vehicular ad hoc networks, smart grid and cloud computing, and remote user authentication. He has authored over 120 papers in international journals and conferences in the above areas. He was a recipient of the Institute Silver Medal from IIT Kharagpur. He is in the editorial board of KSII Transactions on Internet and Information Systems, and the International Journal of Internet Technology and Secured Transactions (Inderscience), and a Guest Editor for the Computers and Electrical Engineering (Elsevier) for the special issue on Big data and IoT in e-healthcare, and has served as a Program Committee Member in many international conferences. For more details, please visit https://sites.google.com/site/iitkgpakdas/.