Wireless Pers Commun (2016) 86:1095–1119
DOI 10.1007/s11277-015-2975-0

Design and Analysis of a Provably Secure Multi-server Authentication Scheme

Dheerendra Mishra[1]

Published online: 1 August 2015
© Springer Science+Business Media New York 2015

Abstract Authenticated key agreement protocols play an important role to ensure authorized and secure communication over public network. In recent years, several authentication protocols have been proposed for single-server environment. Most of these protocols present efficient and secure solution for single-server environment. However, adoption of these protocols for multi-server environment is not feasible as user have to register on each server, separately. On the contrary, multi-server authentication schemes require single registration. The one time registration mechanism makes the system user-friendly and supports inter-operability. Unfortunately, most of the existing multi-server authentication schemes require all servers to be trusted, involvement of central authority in mutual authentication or multiple secret keys. In general, a servers may be semi-trusted, thus considering all server to be trusted does not seems to be realistic scenario. Involvement of central authority in mutual authentication may create bottleneck scenario for large network. Also, computation of multiple secret keys may not be suitable for smart card based environment as smart card keeps limited storage space. To overcome these drawbacks, we aim to design an authentication scheme for multi-server environment, where all servers does not need to be trusted, central authority does not require in mutual authentication and smart card need not to store multiple secret keys. In this paper, we first analyze the security of recently proposed Yeh's smart card based multi-server authentication scheme (Yeh in Wirel Pers Commun 79(3):1621–1634, 2014). We show that Yeh's scheme does not resist off-line password guessing attack, insider attack and user impersonation attack. Furthermore, we propose an efficient multi-server authentication scheme which does not require all servers to be trusted, central authority no longer needed in authentication and smart card need not to store multiple secret keys. We prove the correctness of mutual authentication of our scheme using the widely-accepted BAN logic. Through the security analysis, we show that our scheme is secure against various known attacks

✉ Dheerendra Mishra
dheerendra.m@gmail.com; dheerendra.mishra@lnmiit.ac.in

1 Department of Mathematics, The LNM Institute of Information Technology, Jaipur, India

including the attacks found in Yeh's scheme. In addition, the proposed scheme is comparable in terms of the communication and computational overheads with related schemes.

Keywords Multi-server authentication · Smart card · Security · Privacy

1 Introduction

Advances in communication technology have enhanced the quality of online services. Existing network technology presents a unique platform for various service based industries, where a remote user can access the faster services without moving from his place. In other words, a user can access the remote server at any time and from anywhere over the network. In general, user communicates with the server over the public network, where an active adversary can eavesdrop, intercept, delete, modify or replay the transmitted messages [1, 2]. As a result, an adversary can easily manipulate the communication and can acquire confidential information. To counter these problems, authentication protocols have been designed, which try to ensure secure and authorized communication over public network [3]. In recent years, authentication schemes using smart card have gained significant attention due their applicability and user-friendliness in distributed network environment. This mechanism allows a user to securely access the remote services by using only one memorable password and one legitimate smart card. In other words, input of only a memorable password along with legitimate smart card is enough to initiate the authorized session.

Design and analysis of secure and efficient authentication scheme is an important problem for both system security and network security. The authentication schemes are primarily used for authorized communication between the targeted server and remote user [4–7]. In general, authentication schemes support mutual authentication and session key agreement, where involve entities can verifies the correctness of each other and draw a common key with shared secrets such that only the authorized participants can construct [8, 9]. The established key is called session key, which can be used to transmit confidential data [10, 11]. The existing smart card-based authentication schemes for two party systems provides an efficient and secure way to identify the correctness of participants for single server environment. But, two party authentication schemes do not give scalable solution for multi-server environment for large network due to the following facts:

- A user has to register with each server, independently.
- For each registered server, a user has to maintain the secret key.

If a system has multiple servers, then with multiple authorities a user has to interact to complete the registration. Moreover, a user needs to maintain multiple secret keys in order to access each server separately. However, several multi-server based applications are arising due to widespread distribution of the remote systems, where the traditional two-party architecture is not sufficient to present scalable solution. This may decrease the adoption of network based applications. On the contrary, multi-server based authentication schemes present an efficient solution with the following merits [12].

- Registration with a central authority is sufficient to access multiple servers in the system.
- A unique key could be used to access multiple servers.

- It supports inter-operability.

In recent years, several multi-server based authentication schemes have been proposed in the literature. In 2001, Li et al. [13] introduced multi-server authentication scheme without any verification table. Later, Lin et al. [14] pointed out some drawbacks of Li et al.'s scheme. They showed that Li et al.'s scheme takes long time to train neural networks. To conquer this drawback, they proposed a discrete logarithm problem based multi-server authentication scheme. However, Cao and Zhong [15] identified that Lin et al.'s scheme does not withstand impersonation attack. To gain efficiency in multi-server authentication mechanism, Juang [16] proposed authentication scheme using symmetric-key cryptosystem. Unfortunately, their scheme does not resist insider attack. Chang and Lee [17] also introduced lightweight multi-server authentication scheme. Their scheme is a more efficient than the Juang's scheme. However, Chang and Lee's scheme requires all involved servers to be trusted in the system. Later on, Tsai [18] pointed out that considering the all involved servers in the system to be trusted, may involves several security problems. To ensure security in semi-trusted environment, Tsai proposed a smart-card-based multi-server authentication scheme. His scheme requires the computation of only one-way hash function, and does not require any verification table at the registration center as well as the servers. However, Chen et al. [19] pointed out the vulnerability of Tsai's scheme to server spoofing attack. Tsaur et al. [20] also proposed a self-verified timestamp based authentication scheme for multi-server environments. Unfortunately, their scheme is vulnerable to insider attack and smart card lost password guessing attack [21]. Moreover, none of these multi-server authentication schemes [13, 14, 16–18, 20] protect user anonymity.

To achieve anonymous authentication in multi-server environment, Liao and Wang [22] proposed a dynamic ID-based remote user authentication scheme for multi-server environment. Although Chen et al. [23] identified that Liao–Wang's scheme does not provide forward secrecy. Later on, Hsiang and Shih [24] also pointed out the vulnerability of Liao–Wang's scheme to insider attack, user masquerade attack and server spoofing attack. Moreover, to overcome the failings found in Liao and Wang's scheme, they proposed an improved multi-server authentication scheme. Subsequently, Lee et al. [25] demonstrated vulnerability of Hsiang and Shih's scheme to server spoofing attack. Lee et al. then presented an enhanced multi-server authentication scheme. However, Truong et al. [26] pointed out that Lee et al.'s scheme does not withstand stolen smart card attack and user impersonation attack. They also proposed a revised scheme in order to overcome weaknesses found in Lee et al.'s scheme. Unfortunately, their scheme is vulnerable to insider attack. Sood et al. [27] also identified the weaknesses of Hsiang-Shih's scheme, and proposed an enhanced dynamic ID-based multi-server authentication scheme. Later on, Li et al. [28] pointed out the vulnerability of Sood et al.'s scheme to stolen smart card attack. They also proposed a smart-card based multi-server authentication scheme. Unfortunately, their scheme requires the involvement of a control server in order to achieve mutual authentication between user and server. This may create bottleneck situation for large network. Recently, He and Wang [29] also proposed a multi-server authentication scheme, however, their scheme also requires the involvement of a control server in mutual authentication between user and targeted server. To achieve mutual authentication without the involvement of control server, Wang and Ma [30] introduced a modified password-based authentication scheme for multi-server environment. Later, He and Wu [31] identified that Wang et al.'s scheme is vulnerable to server spoofing attack, privileged insider attack, user impersonation attack and off-line password guessing attack. Recently, Pippal et al. [32] proposed a multi-server authentication scheme using smart-card. Their scheme does not require control server in mutual authentication. However, He et al. [33] demonstrated that Pippal et al.'s scheme does not resist user impersonation attack, privileged insider attack, server spoofing attack and off-line password guessing attack. Yeh [34] also demonstrated the vulnerability of Pippal et al.'s scheme to user impersonation attack and man-in-the-middle attack. To conquer these weaknesses, Yeh proposed a modified multi-server authentication scheme. Unfortunately, we identify that Yeh's scheme does not resist insider attack, password guessing attack and user impersonation attack. Moreover, their scheme require the storage of distinct key for each targeted server in the smart card which makes impossible to add sever on letter stage, otherwise existing users cannot access the services.

Motivation Most of the multi-server authentication schemes either require all servers to be trusted or need to generate multiple secret keys for each user corresponding to each existing server or required trusted central authority in order to achieve mutual authentication. However, for the large network, all the scenarios are infeasible. Assumption of all servers to be trusted is not realistic as a server may be semi-trusted or a server may be compromised. The computation and storage of multiple secret keys in order to access different targeted servers is also not scalable as smart card keeps limited storage space and server cannot be added at later stage. If any server will be added at later stage, the existing users cannot get the services of newly added server. On the other hands, the use of central authority in mutual authentication between user and server can ensure the correct mutual authentication in semi-trusted environment without storing multiple keys in the smart card. However, use of central authority at each instance, may create bottleneck situation for large network. Thus, to achieve efficiency and scalability, an authentication scheme for multi-server environment should meet the following requirement:

- To avoid bottleneck situation in the system, the central authority should be avoided in mutual authentication.
- To design low cost smart card, the storage of multiple secret keys in the smart card should not be required.
- Server can be added on later stage.
- Do not require all involved servers to be trusted.

Our contributions The main contributions of this paper are the following:

- We analyze the security of recently proposed Yeh's multi-server authentication scheme, and show it's vulnerability to off-line password guessing attack and user impersonation attack.
- To overcome the weaknesses of earlier proposed schemes, we present a new dynamic ID-based multi-server authentication scheme where central authority participation is not needed in mutual authentication.
- The proposed scheme ensures anonymity with un-traceability.
- The proposed scheme supports mutual authentication and session key agreement. The correctness of mutual authentication is shown using the BAN logic.
- The proposed scheme is also suitable for semi-trusted servers environment, and do not require the storage of multiple secret keys in user smart card.
- The proposed scheme is resilient to replay attack, man-in-the middle attack and impersonation attack.
- The proposed scheme also withstand off-line password guessing attack, insider attack and stolon smart card attack.
- We also prove the security of our scheme in random oracle model.

- The costs are comparable to the existing approaches.

Organization of the article The rest of the article is arranged as follows: The next Section is preliminary section. Section 3 presents brief review of Yeh's scheme. Subsequently, we discuss the security failure of Yeh's scheme in Sect. 4. Then, we propose our new multi-server authentication scheme in Sect. 5. Correctness of mutual authentication is demonstrated in Sect. 6. Security analysis is given in Sect. 7. The performance is discussed in Sect. 8. We draw the conclusion in Sect. 9.

2 Mathematical Preliminaries

In this section, we briefly discuss some basic mathematical preliminaries in the following subsections for describing and analyzing Yeh's scheme and our scheme.

2.1 Notations

The notation used in the proposed scheme and Yeh's scheme are discussed in Table 1.

2.2 BAN Logic

BAN logic [35, 36] is applied to show the correctness of mutual authentication between the user and server. Using BAN logic, one can demonstrate whether the exchanged information between the user and server is secured and trustworthy or not. It comprises the verification of message origin, message freshness and the message origin. Some notations used in BAN logic analysis are described as follows:

Table 1 Meaning of symbols using throughout the paper

Notation	Descryption
U_i	User i
RC	A trustworthy registration center
S_j	jth server in the system
ID_i	Unique identity of user i
PW_i	Unique password of user i
T_i	Timestamp generated by entity i
SK_{ij}	Session key between user i and server j
mk	Master key of RC
sk_{U_i}/pk_{U_i}	Secret/public key of user i
$h(\cdot), h_1(\cdot)$ and $h_2(\cdot)$	One-way hash functions
p	A large prime
$E_p(a,b)$	An elliptic curve $y^2 = x^3 + ax + b\,(\text{mod p})$ over a finite field Z_p with $4a^3 + 27b^2 \neq 0\,(\text{mod p})$
G	Additive group of points of $E_p(a,b)$, whose order is n
$\oplus$	XOR
$\|\|$	String concatenation operation

- $P \models X$: The principal P believes the statement X.
- $P \triangleleft X$: P sees X, means that P has received a message combine X.
- $P \mid\sim X$: P once said X, means that $P \models X$ when P sent it.
- $P \mid\Rightarrow X$: P controls X, P has an authority on X (Jurisdiction over X).
- $\sharp(X)$: The message X is fresh.
- $P \models Q \overset{k}{\longleftrightarrow} P$: P and Q use K (shared key), to communicate with each other.
- $A \overset{x}{\longleftrightarrow} B : x$ is a shared secret information between A and B.
- $\{X\}_K$: The formula X is encrypted under k.
- $\langle X \rangle_Y$: The formula X is combined with formula Y.
- $(X)_K$: The formula X is hashed with the key K.
- $\overset{k}{\rightarrow} P$: K is public key of P.
- $P \overset{X}{\rightleftharpoons} Q$: X is a secret formula, known only to P and Q.

In order to describe logical postulates of BAN logic in formal terms [35, 37], following rules are defined below:

Rule (1): Message meaning rule:

$$\frac{P \models Q \overset{K}{\leftrightarrow} P, P \triangleleft \{X\}_k}{P \models Q \sim X} \tag{1}$$

If P believes that K is shared with Q and sees X encrypted under k, then P believes that Q once said X.

Rule (2): The nonce verification rule:

$$\frac{P \models \sharp(X), P \models Q \mid\sim X}{P \models Q \models X} \tag{2}$$

If P believes that X has been uttered recently (freshness) and P believes that Q once said X, and then P believes that Q believes X.

Rule (3): The jurisdiction rule:

$$\frac{P \models Q \models X, P \models Q \mid\Rightarrow X}{P \models X} \tag{3}$$

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

Rule (4): The freshness rule:

$$\frac{P \models \sharp(X)}{P \models \sharp(X, Y)} \tag{4}$$

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

2.3 Collision-Resistant One-Way Hash Function

A collision-resistant one-way hash function is defined in [38, 39] as follows.

Definition 1 (*Collision-resistant one-way hash function*) A collision-resistant one-way hash function $h : X \rightarrow Y$, where $X = \{0, 1\}^*$ and $Y = \{0, 1\}^n$ is a deterministic algorithm that takes an input as an arbitrary length binary string $x \in \{0, 1\}^*$ and outputs a binary string $y \in \{0,1\}^n$ of fixed-length n. If we denote $Adv_{\mathcal{A}}^{HASH}(t)$ as an adversary $\mathcal{A}$'s advantage in finding collision, we then have

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

where $Pr[E]$ denotes the probability of a random event E, and $(x,x') \in_R \mathcal{A}$ denotes the pair (x,x') is selected randomly by $\mathcal{A}$. In this case, the adversary $\mathcal{A}$ is allowed to be probabilistic and the probability in the advantage is computed over the random choices made by the adversary $\mathcal{A}$ with the execution time t. We call such a hash function $h(\cdot)$ is collision-resistant, if $Adv_{\mathcal{A}}^{HASH}(t) \leq \epsilon_1$, for any sufficiently small $\epsilon_1 > 0$.

2.4 Elliptic Curve Over a Prime Field

A non-singular elliptic curve $y^2 = x^3 + ax + b$ over the finite field $GF(p)$ is considered as the finite set $E_p(a,b)$ of solutions $(x,y) \in Z_p \times Z_p$ to the congruence $y^2 = x^3 + ax + b \pmod p$, where $a,b \in Z_p$ are constants chosen such that the condition $4a^3 + 27b^2 \neq 0 \pmod p$ is satisfied, together with a special point $\mathcal{O}$ called the point at infinity or zero point, where $Z_p = \{0,1,\ldots,p-1\}$ and $p > 3$ be a prime. The total number of points on the elliptic curve $E_p(a,b)$, which is denoted by $|E|$, satisfies the inequality [40]: $p + 1 - 2\sqrt{p} \leq |E| \leq p + 1 + 2\sqrt{p}$. Thus, we can say that an elliptic curve $E_p(a,b)$ over Z_p has roughly p points. Furthermore, $E_p(a,b)$ forms an commutative group under addition modulo p operation.

Let G be the base point on $E_p(a,b)$, whose order be n, that is, $nG = G + G + \cdots + G(n\,times) = \mathcal{O}$. Assume that $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ are two points on elliptic curve $y^2 = x^3 + ax + b \pmod p$. Then $R = (x_R, y_R) = P + Q$ is computed as follows [41]:

$$\begin{aligned} x_R &= (\gamma^2 - x_P - x_Q) \pmod p, \\ y_R &= (\gamma(x_P - x_R) - y_P) \pmod p, \end{aligned}$$
$$\text{where } \gamma = \begin{cases} \dfrac{y_Q - y_P}{x_Q - x_P} \pmod p, & \text{if } P \neq Q \\ \dfrac{3{x_P}^2 + a}{2y_P} \pmod p, & \text{if } P = Q. \end{cases}$$

In elliptic curve cryptography, multiplication is defined as the repeated additions. For example, if $P \in E_p(a,b)$, then $5P$ is computed as $5P = P + P + P + P + P \pmod p$.

Definition 2 (*Elliptic curve computational Diffie–Hellman (EC-CDH) assumption*) The problem of computing $Q = qP$ and $R = rP$ are relatively easy for given scalar $q, r \in Z_p$ and an elliptic curve point $P \in E_p(a,b)$. However, given two points qP and rP, it is a computationally hard to derive qrP. This problem is called the elliptic curve Computational Diffie–Hellman [41].

This can be defined more formally by considering an Experiment $Exp_G^{cdh}(\mathcal{A})$ where we choose two values q and r in Z_p, compute qP and rP, and then provide qP and rP to $\mathcal{A}$. $\mathcal{A}$ experiment $Exp_G^{cdh}(\mathcal{A})$ outputs 1 if $Z = qrP$ and 0 otherwise. We defined $Adv_G^{cdh}(\mathcal{A}) = Pr[Exp_G^{cdh}(\mathcal{A}) = 1]$ as the advantage of adversary in violating the CDH assumption. The advantage function of the group, $Adv_G^{cdh}(t) = max_{\mathcal{A}}\{Adv_G^{cdh}(\mathcal{A})\}$ with time complexity at most t.

2.5 Threat Model

In this paper, we adopts the following threat model with widely accepted security assumptions about the capacity of adversary and smart card security in password based authentication schemes. The following assumptions are [42–46]:

- The user holds the uniformly distributed low-entropy password from the small dictionary. The server keeps the private key. At the time of registration, the server embeds the personalized security parameters in the smart card.
- An adversary and a participants interact by executing oracle queries that enables an adversary to perform various attacks on authentication protocols.
- The communication channel is controlled by the adversary who has the capacity to intercept, modify, delete, resend and reroute the eavesdropped messages.
- An adversary may steel user's smart card and may extract the stored information from the smart card.

In the password authentication protocol Π, each participant is either a user $U_i \in \mathcal{U}$ or a trusted server S interact number of times. Only polynomial number of queries occurs between adversary and the participants interaction. This enables an adversary to simulates a real attack on the authentication protocol. The possible oracle queries are as follows:

Execute$\left(\Pi_U^i, \Pi_S^j\right)$:	This query models passive attacks against the protocol which is used to simulate the eavesdropping honest execution of the protocol. It prompts an execution of the protocol between the user's instances Π_U^i and server's instances Π_S^j that outputs the exchanged messages during honest protocol execution to $\mathcal{A}$.
Send$\left(\Pi_U^i, m\right)$:	This query sends a message m to an instance Π_U^i, enabling adversary $\mathcal{A}$ for active attacks against the protocol. On receiving m, the instance Π_U^i continues according to the protocol specification. The message output by Π_U^i, if any, is returned to $\mathcal{A}$.
Reveal$\left(\Pi_U^i\right)$:	This query captures the notion of known key security. The instance Π_U^i, upon receiving the query and if it has accepted, provides the session key, back to $\mathcal{A}$.
Corrupt$\left(\Pi_U^i, m\right)$:	These queries together capture the notion of two-factor security. The former returns the password of U_i while the latter returns the information stored in the smart card of U_i.
Test(Π_U^i):	This query is used for determining whether the protocol achieves authenticated key exchange or not. If Π_U^i has accepted, then a random bit $b \in \{0, 1\}$ chosen by the oracle, $\mathcal{A}$ is given either the real session key if $b = 1$, otherwise, a random key drawn from the session-key space.

We say that an instance Π_U^i is said to be open if a query Reveal (Π_U^i) has been made by adversary, and unopened if it is not opened. We say that an instance Π_U^i has accepted if it goes into an accept mode after receiving the last expected protocol message.

Definition 3 Two instances Π_U^i and Π_S^i are said to be partnered if the following conditions hold:

1. Both Π_U^i and Π_S^i accept;
2. Both Π_U^i and Π_S^i share the same session identifications (sid);
3. The partner identification for Π_U^i and Π_S^i and vice-versa.

Definition 4 We say an instance Π_U^i is considered fresh if the following conditions are met:

1. It has accepted;
2. Both Π_U^i and its partner Π_S^i are unopened;
3. They are both instances of honest clients.

Definition 5 Consider an execution of the authentication protocol Π by an adversary $\mathcal{A}$, in which the latter is given access to the Execute, Send, and Test oracles and asks at most single Test query to a fresh instance of an honest clints. Let b' be his output, if $b' = b$, where b is the hidden bit selected by the Test oracle. Let D be user's password dictionary with size |D|. Then, the advantage of $\mathcal{A}$ in violating the semantic security of the protocol Π is defined more precisely as follows:

$$Adv_{\Pi,D}(\mathcal{A}) = |2Pr[b' = b] - 1|$$

The password authentication protocol is semantically secure if the advantage $Adv_{\Pi,D}(\mathcal{A})$ is only negligibly larger than $O(q_s)/|D|$, where q_s is the number of active sessions.

3 Review of Yeh's Multi-server Authentication Scheme

In this section, we briefly discuss Yeh multi-server authentication scheme. It comprises of trusted registration center RC. RC first selects two 1024-bits prime numbers, say p and q. RC chooses a generator $g \in Z_N^*$ and computes $N = p \times q$, where $Z_N^* = (g| 1 \leq g \leq N-1, gcd(g,N) = 1)$. Then, RC generates k random numbers $(r_1, r_2, \cdot, r_k)$ for k servers, where $gcd(r_i, r_j) = 1, gcd(r_i, \phi(N)) = 1$, for $1 \leq i,j \leq k, i \neq j$.

3.1 Server Registration Phase

The server S_j submits identity SID_j to RC over a secure channel. RC assigns r_j to S_j and computes $h(r_j||y_{RC})$, where y_{RC} is a secret value chosen by RC. Finally, RC sends $\{r_j, t, h(r_j||y_{RC}), g, N, h(\cdot)\}$ to S_j via secure channel, where $t = \frac{1}{g^{\prod_{i=1}^{k} r_i} modN}$.

3.2 User Registration Phase

The user U_i submits UID_i, PW_i to RC. Then, RC generates a random number r_i and computes $V = h(UID_i||PW_i||r_i)$. RC issues a smart card SC_i to U_i by including the parameters $\{(s_{1,i}, s_{2,i}, \ldots s_{k,i}), r_i, g, N, V, h(\cdot)\}$, where $s_{j,i} = g^{h(SID_j||UID_i||h(r_j||y_{RC}))\times \prod_{i=1,i\neq j}^{k} r_j} modN$.

Note It is noted that registration center stores multiple secret keys in the smart card. In other words, user needs distinct key to establish a authorized session for each targeted server in the system. For large network, smart card needs to store many keys. Additionally, if any server is added, then all users smart card need to be updated, otherwise existing user cannot access the newly added services.

3.3 Login and Authentication Phase

The registered user can established an authorized session with the desirable server in the system as follows:

- U_i inserts his smart card SC_i in the card reader and inputs his identity UID_i' and password PW_i'. SC_i checks $V \overset{?}{=} h(UID_i||PW_i'||r_i)$. If verification succeeds, U_i is authorized. Then, SC_i generates a random nonce u and computes $M_1 = (s_{j,i} \times g^u) modN$, and then sends $\{UID_i, M_1\}$ to S_j.
- On receiving the message $\{UID_i, M_1\}, S_j$ verifies UID_i. If UID_i is valid, S_j generates a random nonce b and computes B, K, SK_{ij} and M_2, and then sends the message $\{B, M_2\}$ to U_i, where $B = g^{b\times h(r_j||y_{RC})} modN, K = g^{u\times b\times h(r_j||y_{RC})} modN, SK_{ij} = h(K||UID_i||SID_j)$ and $M_2 = h(K||UID_i||SID_j||B||SK_{ij})$.
- On receiving the message $\{B, M_2\}, U_i$ computes $K = (B)^u modN$ and $SK_{ij} = h(K||UID_i||SID_j)$. Then, U_i verifies $M_2 \overset{?}{=} h(K||UID_i||SID_i||A||B||SK_{ij}$. If verification succeeds, U_i computes $M_3 = h(K||UID_i||SID_i||A||B||SK_{ij})$ and sends M_3 to S_j. On receiving M_3, S_j verifies $M_3 \overset{?}{=} h(K||UID_i||SID_j||A||B||SK_{ij})$. If verification succeeds, mutual authentication is achieved. Then, U_i and S_j agree upon a common key SK_{ij}.

4 Cryptanalysis of Yeh's Multi-server Authentication Scheme

In this section, we point out the security flaws found in Yeh's multi-server authentication scheme.

4.1 Off-Line Password Guessing Attack

The password guessing attack is the one of the most common attack on the password-based authentication protocols using smart card. Unfortunately, Yeh's scheme does not resist password guessing attack. An adversary $\mathcal{A}$ can successfully perform the off-line password guessing attack on Yeh's scheme as follows:

Step 1 $\mathcal{A}$ can retrieve the username UID_i of a legal user U_i from the transmitted login message $\{UID_i, M_1\}$ as the adversary can intercept the the messages transmitting via public channel.

Step 2 $\mathcal{A}$ can retrieve V and r_i from the stolen smart card of U_i using the power analysis attack [47, 48].

Step 3 $\mathcal{A}$ can guess a password PW_i^* and compute $V^* = h(UID_i||PW_i^*||r_i)$. Then, $\mathcal{A}$ verifies the condition $V \overset{?}{=} V^*$.

Step 4 If the above verification holds, the guessed password PW_i^* is considered as the correct password PW_i of the user U_i. Otherwise, $\mathcal{A}$ needs to repeat from Step 3 to guess another password.

In is thus clear that Yeh's scheme cannot resist the off-line password guessing attack as a result low-entropy password can be easily guessed.

4.2 Insider Attack

The analysis of the privileged-insider attack on Yeh's scheme shows that an malicious insider can achieve user's password in Yeh's scheme. During the registration phase of Yeh's scheme, a legal user U_i submits his identity ID_i along with password PW_i to the registration center RC via a secure channel. The user submits his original password without masking. This makes possible to an insider to get user's password. Using the password, a malicious insider can access user all those accounts which are protected with the same passwords. However, an efficient scheme should protect user password from insider threat where as Yeh's scheme fails.

4.3 User Impersonation Attack

On Yeh's scheme using stolen smart card, an active adversary $\mathcal{A}$ can easily perform user impersonation attack by masquerading as a legitimate user U_i even without knowing user password PW_i as follows:

- $\mathcal{A}$ uses transmitted messages $\{UID_i, M_1\}$ to retrieve user identity UID_i.
- $\mathcal{A}$ can retrieve $s_{1,i}, s_{2,i}, \ldots, s_{k,i}$ from the stolen smart card of U_i using the power analysis attack [47, 48].
- $\mathcal{A}$ generates a random nonce a and computes $M_a = (s_{j,i} \times g^a)\, mod\, N$, and then sends $\{UID_i, M_a\}$ to S_j.
- On receiving the message $\{UID_i, M_a\}, S_j$ verifies UID_i. The verification of UID_i holds as $\mathcal{A}$ uses registered user U_i's identity. Then, S_j generates a random nonce b and computes B, K, SK_{ij} and M_b, and then sends the message $\{B, M_b\}$ to U_i, where $B = g^{b \times h(r_j || y_{RC})}\, mod\, N, K = g^{a \times b \times h(r_j || y_{RC})}\, mod\, N, SK_{ij} = h(K||UID_i||SID_j)$ and $M_2 = h(K ||UID_i||SID_j||B||SK_{ij})$.
- $\mathcal{A}$ intercepts the message and gets $B = g^{b \times h(r_j || y_{RC})} mod\, N$. $\mathcal{A}$ computes $K = (B)^a modN$ and the session key $SK_{ij} = h(K||UID_i||SID_j)$.

The above discussion shows that an adversary with stolen smart card can correctly masquerade as a legitimate user, and draw the established session key with the server.

4.4 User Anonymity

The leakage of a user's specific information enables an adversary to collect user's login history. On the other hand, the user anonymity prevents an attacker from acquiring the user's sensitive personal information. Moreover, user anonymity makes the remote user authentication mechanism more robust as an attacker could not track which user and server are interacting. Unfortunately, from login message in Yeh's scheme, an adversary can retrieve user's identity and can collect login history. It may cause identity theft. Additionally, using user's identity, an adversary can perform off-line password guessing attack and user impersonation attack as discussed in Sects. 4.1 and 4.3, respectively.

5 Design of a New Multi-server Authentication Scheme

In this section, we proposed an new multi-server authentication scheme. In the proposed scheme, the user does not need to maintain several secret keys in order to login to the distinct servers. Additionally, assumption of all server to be trusted is also not required.

The scheme is designed in such a way that a user and server can efficiently established an authorized session without the involvement of registration center. The proposed scheme comprises of following phases:

- System setup phase
- Server registration phase
- User registration phase
- Authorization phase

5.1 System Setup Phase

In the beginning of the system, the registration center RC selects its public and private parameters.

Step 1 RC selects an elliptic curve $E_p(a, b)$ over a finite field, and chooses P as the generator of the additive group G consisting of the points of $E_p(a, b)$ having the order n.

Step 2 RC selects three hash functions $h : \{0, 1\}^* \to \{0, 1\}^k, h_1 : \{0, 1\}^* \times G \to \{0, 1\}^*$ and $h_2 : \{0, 1\}^* \times \{0, 1\}^* \times \{0, 1\}^* \times G \times G \times G \to \{0, 1\}^k$.

Step 3 RC chooses the master key mk of size 1024-bits. The long key is selected to resist key guessing attack. RC then publishes the system parameters $\{E_p(a, b), P, h(\cdot), h_1(\cdot), h_2(\cdot)\}$, and keeps mk secret.

5.2 Server Registration Phase

In this phase, each server registers and achieves the security parameters. The description is given below:

Step 1 The server S_j submits identity SID_j to RC.

Step 2 RC computes $sk_{S_j} = h(SID_j||mk)$ and $pk_{S_j} = sk_{S_j}P$. RC provides keys (sk_{S_j}, pk_{S_j}) to S_j via a secure channel. RC stores (SID_j, pk_{S_j}) in its public database.

5.3 User Registration Phase

In this phase, registration process of a new user U_i is discussed. A new user achieves his personalized smart card from the registration center which he/she can use to login to the targeted server in the system. For registration, user first selects uniformly distributed identity and password, and then submits identity and masked password with registration request to the registration center via secure channel. Upon receiving the registration request, registration center registers the user, and issues the personalized smart card to him. Additionally, it provides registered users information to all the registered servers in the system. The summery of mechanism is given in Fig. 1.

Step 1 U_i chooses identity UID_i and password PW_i. U_i computes $RPW_i = h(PW_i||UID_i)$. Then, U_i submits $RPW_i \oplus N$ and UID_i to RC via a secure channel, where N is randomly selected value.

Step 2 RC computes $sk_{U_i} = h(UID_i||mk)$ and $X_i = sk_{U_i} \oplus RPW_i \oplus N$, and stores $X_i, P, h(\cdot), h_1(\cdot), h_2(\cdot)$ in the memory of the smart card SC_i. Then, RC provides the smart card SC_i to U_i via a secure channel.

User (U_i)	Secure channel	Registration center (RC)
Choose $UID_i \& PW_i$		
Compute $RPW_i = h(PW_i \vert\vert UID_i)$		
Randomly select N	$\xrightarrow{\text{REQUEST} < RPW_i \oplus N, UID_i >}$	Compute $sk_{U_i} = h(UID_i \vert\vert mk)$
		Compute $X_i = sk_{U_i} \oplus RPW_i \oplus N$
	$\xleftarrow{SC_i\{X_i, P, h(\cdot), h_1(\cdot), h_2(\cdot)\}}$	Store $X_i, h(\cdot), h_1(\cdot), h_2(\cdot)$, & P into SC_i
Compute $Y_i = X_i \oplus N$		Compute $pk_{U_i} = sk_{U_i} P$
Replace X_i with Y_i		Provide (UID_i, pk_{U_i}) to all S_j

Fig. 1 Summary of user registration

Step 3 RC also computes $pk_{U_i} = sk_{U_i}P$, and provides (UID_i, pk_{U_i}) to all the registered servers in the system. All servers stores (UID_i, pk_{U_i}).

Step 4 On receiving the smart card, U_i computes $Y_i = X_i \oplus N$, and replaces X_i with Y_i.

5.4 Authorization Phase

In order to access the services of a targeted server S_j, a valid user U_i first initiates the authorized session with S_j. For secure and authorized communication, the user and server verify the correctness of each other, and then draw a common key without any involvement of trusted registration center. The description of authorization phase is given below. The summary of authorization phase is given in Fig. 2.

Step 1 U inserts his smart card SC_i into a card reader, and inputs his identity UID_i and password PW_i. Then, user selects the targeted server S_j for the services in the system, and achieves the public credentials (SID_j, pk_{S_j}) from the registration center public directory.

Step 2 SC_i generates a random number u, and computes $V_i = uP, k_{ij} = sk_{U_i}pk_{S_j} = sk_{U_i}sk_{S_j}P$ and $z_{ij} = upk_{S_j}$. Then, SC_i computes masked identity $DID_i = UID_i \oplus h_1(SID_j||z_{ij})$, and sends the message $REQUEST\langle DID_i, V_i, M_i, T_i\rangle$ to S_j via a public channel, where $M_i = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||V_i)$ and T_i is the current timestamp used by U_i.

Step 3 On receiving the login request $REQUEST\langle DID_i, V_i, M_i, T_i\rangle$ at time T_i', S_j verifies the condition $T_i' - T_i \leqslant \varDelta T$. If verification holds, S_j computes

User (U_i)	Public channel	Server (S_j)
Randomly select u		
Compute $V_i = uP$		
$k_{ij} = sk_{U_i} sk_{S_j} P$ and $z_{ij} = upk_{S_j}$		
$DID_i = UID_i \oplus h_1(SID_j \vert\vert z_{ij})$		
$M_i = h(UID_i \vert\vert SID_j \vert\vert T_i \vert\vert k_{ij} \vert\vert z_{ij} \vert\vert V_i)$	$\xrightarrow{\text{REQUEST}< DID_i, V_i, M_i, T_i >}$	Verify $T_i' - T_i \leqslant \varDelta T$
		Compute $z_{ji} = sk_{S_j} V_i$
		Retrieve $UID_i = DID_i \oplus h_1(SID_j \vert\vert z_{ji})$
		Randomly select s
		Computes $k_{ji} = sk_{S_j} pk_{U_i}$
	$\xleftarrow{\text{AUTHORIZATION}}$	Verify $M_i \overset{?}{=} h(UID_i \vert\vert SID_j \vert\vert T_i \vert\vert k_{ji} \vert\vert z_{ji} \vert\vert V_i)$
		Compute $V_j = sp, w_{ji} = sV_i$
		$SK_{ji} = h(UID_i \vert\vert SID_j \vert\vert T_i \vert\vert k_{ji} \vert\vert z_{ji} \vert\vert w_{ji})$
		$M_j = h(UID_i \vert\vert SK_{ji} \vert\vert T_j \vert\vert k_{ji} \vert\vert w_{ji} \vert\vert V_j)$
Verify $T_j' - T_j \leqslant \varDelta T$	$\xleftarrow{\text{CHALLENGE}\{V_j, M_j, T_j\}}$	
Compute $w_{ij} = uV_j$		
$SK_{ij} = h(UID_i \vert\vert SID_j \vert\vert T_i \vert\vert k_{ij} \vert\vert z_{ij} \vert\vert w_{ij})$		
Verify $M_j \overset{?}{=} h(UID_i \vert\vert SK_{ij} \vert\vert T_j \vert\vert k_{ij} \vert\vert w_{ij} \vert\vert V_j)$	$\xrightarrow{\text{AUTHORIZATION}}$	

Fig. 2 Summary of authorization phase of proposed scheme

$z_{ji} = sk_{S_j}V_i = sk_{S_j}uP$, and then extracts UID_i as $UID_i = DID_i \oplus h_1(SID_j||z_{ji})$. S_j extracts pk_{U_i} stored corresponding UID_i from its database. S_j computes $k_{ji} = sk_{S_j}pk_{U_i} = sk_{S_j}sk_{U_i}P$, and then verifies $M_i \overset{?}{=} h(UID_i||SID_j||T_i||k_{ji}|| z_{ji}||V_i)$. If verification does not hold, the session is immediately terminated. Otherwise, U_i's verification holds.

Step 4 S_j generates a random number s and computes $V_j = sp, w_{ji} = sV_i = sup$ and the session key $SK_{ji} = h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$. Then, S_j responds with the message $CHALLENGE\langle V_j, M_j, T_j\rangle$ to U_i via a public channel, where $M_j = h(UID_i||SK_{ji}||T_j||k_{ji}||w_{ji}||V_j)$ and T_j is the current timestamp used by S_j.

Step 5 On receiving the message $CHALLENGE\langle V_j, M_j, T_j\rangle$ at time T'_j, U_i verifies the condition $T'_j - T_j \leqslant \Delta T$. If verification succeeds, U_i computes $w_{ij} = uV_j$ and $SK_{ij} = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||w_{ij})$. Then, U_i verifies $M_j \overset{?}{=} h(UID_i||SK_{ij}||T_j|| k_{ij}||w_{ij}||V_j)$. If verification succeeds, U_i considers sk_{ij} as the valid common key and S_j as an authorized server.

User and server verify the correctness of authenticity of each other, and after verification they compute the session key which is common at both ends, that is, $SK_{ij} = SK_{ji}$, it is justified below:

$$\begin{aligned} k_{ij} &= sk_{U_i}sk_{S_j}P = sk_{S_j}sk_{U_i}P = k_{ji} \\ z_{ij} &= usk_{S_j}P = sk_{S_j}uP = z_{ji} \\ w_{ij} &= usP = suP = w_{ji} \end{aligned}$$

It is clear that $k_{ij} = k_{ji}, z_{ij} = z_{ji}$ and $w_{ij} = w_{ji}$, and so $SK_{ij} = h(UID_i||SID_i||T_i||k_{ij}||z_{ij}| |w_{ij}) = h(UID_i||SID_i||T_i||k_{ji}||z_{ji}||w_{ji}) = SK_{ji}$.

6 Accuracy of Mutual Authentication

We apply logical postulates of BAN logic [35, 36] to show the correctness of mutual authentication between the remote user and server. Using BAN logic, we show that the user and server can correctly determine correctness of exchanged information over public channel. It comprises the verification of message origin, message freshness and the origin's trustworthiness. In the proposed scheme, the generic form of the messages exchange between the user and server are as follows:

Message 1: $U_i \rightarrow S_j : \langle UID_i \oplus h_1(SID_j||usk_{S_j}P), h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||V_i), uP, T_i\rangle$
Message 2: $S_j \rightarrow U_i : \langle h(UID_i||SK_{ji}||T_j||k_{ji}||w_{ji}||V_j), sP, T_j\rangle$

Subsequently, we translate the message 1 and 2 into idealize form as follows:

Message 1: $U \rightarrow S_j : (UID_i, SID_j, T_i, z_{ij}, uP)_{k_{ij}}, T_i$

Message 2: $S_j \rightarrow U : (U_i \overset{SK_{ij}}{\longleftrightarrow} S_j, UID_i, T_i, T_j)_{suP}, T_j$

Recall that in the proposed scheme, the user and server use fresh timestamp. We make the following assumptions about the initial state of the proposed scheme:

A_1: $U_i \models \sharp(T_i)$;
A_2: $S_j \models \sharp(T_j)$;

A_3: $U_i \mid\equiv (U_i \xleftrightarrow{k_{ij}} S_j)$;

A_4: $S_j \mid\equiv (U_i \xleftrightarrow{k_{ij}} S_j)$;

A_5: $U_i \mid\equiv S_j \mid\equiv (U_i \xleftrightarrow{k_{ij}} S_j)$;

A_6: $S_j \mid\equiv U_i \mid\equiv (U_i \xleftrightarrow{k_{ij}} S_j)$.

Lemma 1 *The server can correctly verify the authenticity of user's message.*

Proof User generates a login message and sends to the server in order to login to the server. With the message, the server receives the timestamp with other values which help to prove the correctness of message source as follows:

S_1: According to the message 1, we could get: $S_j \triangleleft (UID_i, SID_j, T_i, z_{ij}, uP)_{k_{ij}}, T_i$.

S_2: According to the assumption A_4, we apply the message meaning rule to get: $S_j \mid\equiv U_i \mid\sim T_i$.

S_3: According to the assumption A_1, we apply the freshness-propagation rule to get: $S_j \mid\equiv \sharp(UID_i, SID_j, T_i, z_{ij}, uP)_{k_{ij}}$.

S_4: According to the S_2 and S_3, we apply nonce verification rule and obtain: $S_j \mid\equiv U_i \mid\equiv (UID_i, SID_j, T_i, z_{ij}, uP)_{k_{ij}}$.

S_5: According to the assumption A_4 and S_4, we apply the jurisdiction rule and get: $S_j \mid\equiv T_i$.

The server identify that the used timestamp in the message is fresh. This proves the correctness of message source.

Lemma 2 *The user can correctly verify the server's response message authenticity.*

Proof In the proposed scheme, when correctness of user's login message holds, the server responds with a message which includes the server's timestamp. The user can be able to identify the authenticity of server's message as follows:

S_6: According to the message 2, we could obtain: $(U_i \xleftrightarrow{SK_{ij}} S_j, UID_i, T_i, T_j)_{suP}, T_j$.

S_7: According to the assumption A_3, we apply the message meaning rule to get: $U_i \mid\equiv S_j \mid\sim T_j$.

S_8: According to the assumption A_1, we apply the freshness conjuncatenation rule to get: $U_i \mid\equiv \sharp(U_i \xleftrightarrow{SK_{ij}} S_j, UID_i, T_i, T_s)_{suP}$.

S_9: To compute the session key SK_{ij} $(=h(UID_i||SID_i||T_i||k_{ij}||z_{ij}||w_{ij}))$, the shared secret value k_{ij} is needed and so we get: $U_i \mid\equiv \sharp(UID_i, SID_j, T_i, T_j, suP)_{k_{ij}}$.

S_{10}: According to the S_8 and S_9, we apply nonce verification rule to obtain: $U_i \mid\equiv S_j \mid\equiv (UID_i, SID_j, T_i, T_j, suP)_{k_{ij}}$.

S_{11}: According to the assumption A_1, A_3 and S_{10}, we apply the jurisdiction rule to get: $U_i \mid\equiv T_j$.

This shows that the user can correctly verify the correctness of message source and its freshness.

Theorem 1 *The user and server can mutually authenticate each other.*

Proof According to the Lemma 1, the server can correctly verify the login message sender authenticity. According to the Lemma 2, the user can also correctly verify the authenticity of server's response. This shows that the user and server can mutually authenticate each other.

7 Security Analysis

7.1 Formal Security Analysis of the Proposed Scheme

In order to show that the proposed scheme withstand the known attacks of the authentication protocols, we use the method of provable security. The security proof is based on the model of ECC-based password authentication [46, 49, 50].

Theorem 2 *Let D be an uniformly distributed dictionary of possible passwords with size* $|D|$, *Let* Π *be the improved authentication protocol described in Algorithm 1 and 2. Let* $\mathcal{A}$ *be an adversary against the semantic security within a time bound t. Suppose that CDH assumption holds, then,*

$$Adv_{\Pi,D}(\mathcal{A}) = \frac{2q_h^2}{p} + \frac{2q_s}{p} + \frac{(q_s + q_e)^2}{p} + 2q_h Adv_G^{cdh}(\mathcal{A}) + \frac{2q_h}{p} + \frac{2q_s^2}{D}$$

where $Adv_G^{cdh}(\mathcal{A})$ *is the success probability of* $\mathcal{A}$ *of solving the elliptic curve based computational Diffie–Hellman problem.* q_s *is the number of Send queries,* q_e *is the number of Execute queries and* q_h *is the number of random oracle queries.*

Proof This proof defines a sequence of hybrid games, starting at the real attack and ending up in game where the adversary has no advantage. For each game $G_i (0 \le i \le 5)$, we define an event $Succ_i$ corresponding to the event in which the adversary correctly guesses the bit b in the test-query.

Game G_0 This game correspond to the real attack in the random oracle model. In this game, all the instances of U_i and the server S_j are modeled as the real execution in the random oracle. By definition of event $Succ_i$ in which the adversary correctly guesses the bit b involved in the Test-query, we have

$$Adv_{\Pi,D}(\mathcal{A}) = 2\left|Pr[Succ_0] - \frac{1}{2}\right| \tag{5}$$

Game G_1 This game is identical to the game G_0, except that we simulate the hash oracles h by maintaining the hash lists $List_h$ with entries of the form (Inp, Out). On hash query for which there exists a record (Inp, Out) in the hash list, return Out. Otherwise, randomly choose $Out \in \{0, 1\}$, send it to $\mathcal{A}$ and store the new tuple (Inp, Out) into the hash list. The Execute, Reveal, Send, Corrupt, and Test oracles are also simulated as in the real attack where the simulation of the different polynomial number of queries asked by $\mathcal{A}$. From the viewpoint of $\mathcal{A}$, we identify that the game is perfectly indistinguishable from the real attack. Thus, we have

$$Pr[Succ_1] = Pr[Succ_0] \tag{6}$$

Game G_2 In this game, the simulation of all the oracles is identical to game G_1 except that the game is terminated if the collision occurs in the simulation of the transcripts $\langle DID_i, V_i, M_i, T_i \rangle$ and $\langle V_j, M_j, T_j \rangle$. According to the birthday paradox, the probability of collisions of the simulation of hash oracles is at most $\frac{q_h^2}{2p}$. Similarly, the probability of collisions in the transcripts simulations is at most $\frac{(q_h+q_e)^2}{2p}$. Since u and s was selected uniformly at random. Thus, we have

$$|Pr[Succ_2] - Pr[Succ_1]| = \frac{q_h^2}{2p} + \frac{(q_s+q_e)^2}{2p} \tag{7}$$

Game G_3 The simulation of this game is similar to the previous game except the game will be aborted if $\mathcal{A}$ can correctly guessed the authentication values M_i and M_j without asking oracle h. This game and earlier game are indistinguishable unless the instances $\Pi^i_{U_i}$ and $\Pi^i_{S_j}$ rejects a valid authentication value. Hence, we have

$$|Pr[Succ_3] - Pr[Succ_2]| \leq \frac{q_h}{p} \tag{8}$$

Game G_4 In this game, the session key is guessed without asking the corresponding oracle h so that it become independent of password and ephemeral keys usP. We change the way with earlier game unless $\mathcal{A}$ queries h on the common value $h(UID_i||SID_i||T_i||k_{ij}||z_{ij}||usP)$. Thus, $Adv_G^{cdh}(\mathcal{A}) \geq \frac{1}{q_h}|Pr[Succ_4] - Pr[Succ_3]| - \frac{1}{p}$, that is, the difference between the game G_4 and the game G_3 is as follows:

$$|Pr[Succ_4] - Pr[Succ_3]| \leq q_h Adv_G^{cdh}(\mathcal{A}) + \frac{q_h}{p} \tag{9}$$

Game G_5 This game is similar to the game G_4 except that in Test query, the game is aborted if $\mathcal{A}$ asks a hash function query with $h(UID_i||SID_i||T_i||k_{ij}||z_{ij}||usP)$. $\mathcal{A}$ gets the session key SK_{ij} by hash function query with probability at most $\frac{q_h^2}{2p}$. Hence, we have

$$|Pr[Succ_5] - Pr[Succ_4]| \leq \frac{q_h^2}{2p} \tag{10}$$

If $\mathcal{A}$ does not make any h query with the correct input, it will not have any advantage in distinguishing the real session key from the random once. Moreover, if the corrupt query Corrupt (U, 2) is made that means the password-corrupt query Corrupt (U, 1) is not made. Thus, the probability of $\mathcal{A}$ made off-line password guessing attack is $\frac{q_s^2}{D}$. Combining the Eqs. 5–10 one gets the announced result as:

$$Adv_{\Pi,D}(\mathcal{A}) = \frac{2q_h^2}{p} + \frac{2q_s}{p} + \frac{(q_s+q_e)^2}{p} + 2q_h Adv_G^{cdh}(\mathcal{A}) + \frac{2q_h}{p} + \frac{2q_s^2}{D}$$

Algorithm 1 Simulation of send query

```
1: On the query $Send(\Pi_U^i, start)$, assume that $U_i$ is in correct state, then we proceed as follows:
2: Choose a number $u \in_R Z_p^*$, compute $upk_{S_j}$, $DID_i = UID_i \oplus h_1(SID_j||z_{ij})$, $V_i = uP$ and $M_i = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||V_i)$. This query returns $\langle DID_i, V_i, M_i, T_i \rangle$ as answer.
3: On a query $Send(S_j, < DID_i, V_i, M_i, T_i >)$, assume that $S_j$ is in correct state, we continue as follows.
4: if $T_i' - T_i > \Delta T$ then
5:     Reject the message.
6: else Compute $z_{ij} = sk_{S_j}V_i, UID_i = DID_i \oplus h_1(SID_j||z_{ij}), k_{ji} = sk_{S_j}pk_{U_i}$ and $M_i^* = h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||V_i)$
7:     if $M_i^* \neq M_i$ then
8:         Reject the request.
9:     else Compute $V_j = sP, w_{ji} = sV_i, SK_{ji} = h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$ and $M_j = h(UID_i||SK_{ji}||T_j||k_{ji}||w_{ji}||V_j)$. The query returns $< V_j, M_j, T_j >$ as answer.
10:     end if
11: end if
12: On a $Send < V_j, M_j, T_j >$, assume that $U_i$ is in correct state, we continue as follows:
13: if then $T_s' - T_s > \Delta T$
14:     Reject the message $< V_j, M_j, T_j >$.
15: else Compute $w_{ij} = uV_j, SK_{ij} = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||w_{ij})$ and $M_j^* = h(UID_i||SK_{ij}||T_j||k_{ij}||w_{ij}||V_j)$
16:     if $M_j^* \neq M_j$ then
17:         Reject the response.
18:     else The user instance accepts.
19:     end if
20: end if
```

Algorithm 2 Simulation of Execute query

```
On a query $Reveal(\Pi_U^i)$, we proceed as follows:
if The instance $\Pi_U^i$ is accepted then
    This query answered the session key.
end if
```

7.2 Further Security Discussion of the Proposed Scheme

In this section, we discuss that the proposed scheme have all the security feature of session key agreement and functionality of two-factor password-based authentication including user's anonymity.

Proposition 1 *The proposed scheme could provide user's anonymity with un-linkability.*

Proof The login message $\{DID_i, V_i, M_i, T_i\}$ includes DID_i instead of UID_i. To retrieve UID_i from DID_i is equivalent to compute $z_{ij} = usk_{S_j}P$ using uP and pk_{S_j} as $DID_i = UID_i \oplus h_1(SID_j||z_{ij})$. Computation of z_{ij} using uP and pk_{S_j} is equivalent to elliptic curve computational Diffie–Hellman (EC-CDH) problem. As EC-CDH is considered to be a computationally hard problem (defined in Definition 2), the adversary cannot retrieve UID_i from DID_i. Moreover, user randomly chooses value u for each session which makes z_{ij} different for each session so as DID_i. Additionally, no message part is repeated in consecutive communications. This shows that our scheme achieve un-linkability property along with anonymity.

Proposition 2 *The proposed scheme could withstand privileged-insider attack.*

Proof During the registration phase, a legal user U_i submits masked password $RPW_i \oplus N$ to the registration center RC instead of password PW_i, where $RPW_i = h(ID_i||PW_i)$ and N is a randomly selected value. Thus, an insider cannot achieve the password PW_i due to the non-retrieval property of the one-way hash function $h(\cdot)$. Moreover, the insider cannot guess the password as user does not submit N to the server. This shows that the proposed scheme resists insider attack.

Proposition 3 *The proposed scheme could resist stolen verifier attack.*

Proof In proposed scheme, each server store (UID_i, pk_{U_i}). To retrieve, sk_{U_i} from pk_{U_i} is equivalent to CDH problem in ECC. As the server secret key is only known to the server and user stored information does not reveal user secret key, the proposed scheme withstands the stolen verifier attack.

Proposition 4 *The proposed scheme could resist off-line password guessing attack.*

Proof In this attack, an adversary may try to guess a legal user U_i's password PW_i using the transmitted messages. In proposed scheme, the adversary may try to verify the password using the condition $M_i = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||V_i)$ or $M_j = h(UID_i||SK_{ji}||T_j||k_{ji}||w_{ji}||V_j)$ as $SK_{ji} = h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$. However, this attempt does not succeed in the proposed scheme which is justified below:

- To verify the guessed password PW_i^* using $M_i = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||V_i)$, the adversary has to compute upk_{S_j}. To compute upk_{S_j} using uP and pk_{S_j}, is equivalent to EC-CDH assumption.
- To verify the guessed password PW_i^* using $M_j = h(UID_i||SK_{ji}||T_j||k_{ji}||w_{ji}||V_j)$, the adversary has to compute suP. The computation of suP using uP and sP, is equivalent to EC-CDH assumption.

It is clear from the above discussion that guessing password in the proposed scheme is equivalent to elliptic curve discrete logarithm problem, which is hard.

Proposition 5 *The proposed scheme could withstand replay and man-in-the-middle attacks.*

Proof The login and verification messages include the timestamp. Therefore, an adversary cannot repeat the messages, since the maximum transmission delay ΔT is very short in communication. To modify the message $\langle DID_i, V_i, M_i, T_i\rangle$ with another modified message $\{DID_i, V_A, M_A, T_A\}$ for current timestamp T_A, the adversary (A) has to compute M_A which requires the user U's password PW_i and identity ID_i as $M_A = h(UID_i||SID_j||T_A||k_{ij}||z_{ij}||V_A)$ for timestamp T_A and random value a. Since the user's password PW_i is secret, an adversary cannot achieve it. Moreover, to replace $\langle DID_i, V_i, M_i, T_i\rangle$ with $\langle DID_i, V_i, M_i, T_A\rangle$, an adversary has to compute $M_i = h(UID_i||SID_j||T_i||k_{ij}||z_{ij}||V_i)$, which also requires PW_i and UID_i. As only valid user know UID_i and RPW_i, our proposed scheme resists the replay and man-in-the-middle attacks.

Proposition 6 *The proposed scheme could resist user impersonation attack.*

Proof In such an attack, an adversary may try to masquerade as a legitimate user U_i to successfully login to the server S_j. However, our proposed scheme resists this attack.

- The adversary A may try to login to the server S_j using the replay attack. However, the proposed scheme resists the replay attack.
- The adversary A may try to generate a valid login message $\langle DID_i, aP, M_A, T_A\rangle$ for a random value a and current timestamp T_A, where $M_A = h(UID_i||SID_j||T_A||k_{ij}||z_{ij}||aP)$. However, the adversary cannot compute M_A as computation of M_A requires PW_i and UID_i, both values are only known to user.

It is clear that the adversary cannot generate valid login message. This shows that the proposed scheme resist user impersonation attack.

Proposition 7 *The proposed scheme could withstand server impersonation attack.*

Proof In this attack, an adversary can masquerade as the server S_j and try to respond with a valid message to the user U_i. When a user U_i sends a login message $\langle DID'_i, u'P, M'_i, T'_i\rangle$ to the server S_j, the adversary intercepts this message and try to respond with a valid message, where $M'_i = h(UID_i||SID_j||T'_i||k_{ij}||z'_{ij}||u'P)$. However, the proposed scheme resist this attack as follows:

- The adversary may try to respond using the old transmitted message $\langle V_j, M_j, T_j\rangle$ of S_j. This attempt cannot succeed as the login and response message includes timestamp, and proposed scheme resists replay attack.
- The adversary may try to generate a valid response message $\langle aP, M_A, T_A\rangle$ for current timestamp T_A, where $M_A = h(UID_i||SK'_{ji}||T_A||k_{ji}||au'P||aP)$ and $SK'_{ji} = h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||au'P)$. This requires k_{ij} and UID_i.

This shows that our proposed scheme has the ability to resist the server impersonation attack.

Proposition 8 *The proposed scheme could support mutual authentication.*

Proof In our scheme, the server S_j verifies the authenticity of user U_i's request by verifying the condition $M_i \overset{?}{=} h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||V_i)$ during the authorization phase. To compute M_i, U_i's identity UID_i and secret key sk_{U_i} are needed. Therefore, an adversary cannot forge the message. Additionally, M_i includes timestamp, the adversary cannot replay the old message. This shows that the server S_j can correctly verify the message source. U_i also verifies the authenticity of the server S_j with the condition $M_j \overset{?}{=} h(UID_i||SK_{ji}||T_j||k_{ji}||w_{ji}||V_j)$, which also requires UID_i and sk_{U_i}. This shows that the user U_i can also correctly verify the server S_j challenge. Hence, mutual authentication between U_i and S_j can successfully achieved in our scheme.

Proposition 9 *The proposed scheme could have Key freshness property.*

Proof Note that in our scheme, each established session key $SK_{ji} = h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$ includes timestamp T_i, and random values u and s. The timestamp are used to achieve the freshness for each session. Uniqueness property of timestamp, guaranties the unique key for each session. The unique key construction for each session shows that proposed scheme supports the key freshness property.

Proposition 10 *The proposed scheme could have known key secrecy property.*

Proof In our scheme, if a previously established session key $h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$ is compromised, the compromised session key reveals no information about other session keys due to following reasons:

- Each session key is hashed with one-way hash function. Therefore, no information can be retrieved from the session key.
- Each session key includes timestamp, which ensures different key for each session.

Since no information about other established session keys from the compromised session key is extracted, our proposed scheme achieves the known key secrecy property.

Proposition 11 *The proposed scheme could have forward secrecy property.*

Proof Forward secrecy states that compromise of a legal user's long-term secret key does not become the reason to compromise of the established session keys. In our proposed scheme, if the user U_i's user's long-term secret key sk_{U_i} is compromised, an adversary cannot compute the session key due to the following facts:

- To compute the session key SK, user identity UID_i, usP and $usk_{S_j}P$ are needed along with k_{ij} as session key is $h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$.
- Neither the smart card nor anyone of the transmitted messages includes UID_i. So, the adversary cannot derive UID_i.
- The computation of $usk_{S_j}P$ using uP and $sk_{S_j}P$ is equivalent to solve EC-CDH. But, EC-CDH is a computationally hard, the adversary cannot compute umP.
- To compute usP using uP and sP is also equivalent to solve EC-CDH assumption.

This shows that our scheme preserves the forward secrecy property.

Proposition 12 *The proposed scheme could have perfect forward secrecy.*

Proof A scheme is said to support perfect forward secrecy, if the adversary cannot compute the established session key, using compromised secret key sk_{S_j} of any server. The proposed scheme achieves perfect forward secrecy. The description is given below:

- To compute the session key $h(UID_i||SID_j||T_i||k_{ji}||z_{ji}||w_{ji})$, the adversary needs $UID_i, k_{ji}, z_{ij} = upk_{S_j}$ and usP.
- The adversary can compute $z_{ij} = upk_{S_j}P$ using compromised key sk_{S_j} of S_j.
- The adversary can retrieve UID_i as $UID_i = DID_i \oplus h_1(SID_j||z_{ij})$.

The adversary can achieve UID_i, z_{ij} and k_{ji} using compromised key sk_{S_j} of S_j. However, the adversary cannot compute usP as computation of usP from uP and sP is equivalent to solve EC-CDH. This shows that our scheme provides the perfect forward secrecy property.

8 Discussion

In this section, we compare the performance of the proposed scheme with some related multi-server authentication schemes using smart card such as Sood et al.'s scheme [27], Lee et al.'s scheme [25], Li et al.'s scheme [28], Wang and Ma's scheme [30], Pippal et al.'s scheme [32], Yeh's scheme [34].

In Table 2, we compare the computational overhead comparison of proposed scheme and other related schemes. For computing the computational costs of different schemes, we use the following notations. Let $T_{ecm}, T_{eca}, T_h, T_{ex}$ and T_m denote the complexity of executing an elliptic curve point multiplication operation, an elliptic curve point addition, a one-way hash function, modular exponential operation and modular multiplication/inverse operation, respectively. From this, we see that our scheme requires computation cost from user's side and server's side $5T_h + 4T_{ecm}$ and $4T_h + 4T_{ecm}$, respectively.

In Table 3, we compare the communication overhead of proposed scheme with other related schemes, namely Sood et al.'s scheme [27], Lee et al.'s scheme [25], Li et al.'s scheme [28], Wang and Ma's scheme [30], Pippal et al.'s scheme [32], and Yeh's scheme [34] for the login and authentication phases. We assume that the hash digest (output) is 160 bits, if we use SHA-1 hash function [51], timestamp is 32 bits, user identity *username* is 160 bits and random nonce/number is 160 bits. We take prime p for modulo exponential is

Table 2 Computational overhead comparison between our scheme and other schemes

Protocols/overhead	User side	Server side	Total computation
Sood et al.'s protocol [27]	$11T_h$	$11T_h$	$22T_h$
Lee et al.'s protocol [25]	$8T_h$	$7T_h$	$15T_h$
Li et al.'s protocol [28]	$11T_h$	$17T_h$	$28T_h$
Wang and Ma's protocol [30]	$6T_h + 3T_{ecm}$	$5T_h + 3T_{ecm}$	$11T_h + 6T_{ecm}$
Pippal et al.'s protocol [32]	$4T_h + 3T_{ex} + 1T_m$	$3T_h + 4T_{ex} + 1T_m$	$7T_h + 7T_{ex} + 2T_m$
Yeh's protocol [34]	$4T_h + 2T_{ex} + T_m$	$5T_h + 4T_{ex} + T_m$	$9T_h + 6T_{ex} + 2T_m$
Proposed scheme	$5T_h + 4T_{ecm}$	$4T_h + 4T_{ecm}$	$9T_h + 8T_{ecm}$

Table 3 Communication overhead comparison between our scheme and other related multi-server authentication schemes

Scheme	Communication overhead
Sood et al.'s protocol [27]	4 Messages (2240 bits)
Lee et al.'s protocol [25]	3 Messages (1120 bits)
Li et al.'s protocol [28]	4 Messages (2720 bits)
Wang and Ma's protocol [30]	2 Messages (1120 bits)
Pippal et al.'s protocol [32]	3 Messages (5056 bits)
Yeh's protocol [34]	3 Messages (5056 bits)
Proposed scheme	2 Messages (1184 bits)

1024-bits. We take 160-bit ECC cryptosystem. Thus, for an elliptic curve $E_p(a, b)$, each parameter p, a and b requires 160 bits. A point $P = (x_P, y_P) \in E_p(a, b)$ then requires $(160 + 160) = 320$ bits. In our scheme, the REQUEST message $\langle DID_i, V_i, M_i, T_i \rangle$ requires $(160 + 320 + 160 + 32) = 672$ bits, the CHALLENGE message $\langle V_j, M_j, T_j \rangle$ requires $(320 + 160 + 32) = 512$ bits. As a result, our scheme needs $(672 + 512) = 1216$ bits for the communication overhead of two transmitted messages. From Table 3, it is clear that our scheme requires less communication overhead as compared to Sood et al.'s scheme [27], Li et al.'s scheme [28], Pippal et al.'s scheme [32], and Yeh's scheme [34].

Table 4 Features comparison between our scheme and other schemes

Security features	[27]	[25]	[28]	[30]	[32]	[34]	Proposed scheme
User anonymity	Yes	Yes	Yes	No	No	No	Yes
Insider attack	Yes	Yes	Yes	Yes	No	No	Yes
Off-line password guessing attack	No	No	No	Yes	No	No	Yes
Stolen smart card attack	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Denial-of-service attack	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Known session keys attack	Yes	Yes	Yes	Yes	Yes	Yes	Yes
User impersonation attack	No	No	No	No	No	No	Yes
Server impersonation attack	Yes	No	No	No	No	Yes	Yes
Man-in-the middle attack	Yes	No	No	No	No	Yes	Yes
Replay attack	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mutual authentication	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Session key agreement	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Forward secrecy	No	No	Yes	Yes	Yes	Yes	Yes

Finally, in Table 4, we summarize the comparison of security features provided by the proposed scheme and other related schemes, where symbol ’Yes’ used if the protocol support the attribute, otherwise, ‘No’ is used. From Table 4, it is clear that the proposed scheme provides better security features. The proposed scheme has the ability to support other good features such as user’s anonymity and formal security analysis. The scheme is superior in terms of features as compared to relevant multi-server authentication schemes: Sood et al.’s scheme [27], Lee et al.’s scheme [25], Li et al.’s scheme [28], Wang and Ma’s scheme [30], Pippal et al.’s scheme [32], and Yeh’s scheme [34].

9 Conclusion

We have discussed the merits and demerits of the existing password based multi-server authentication schemes in the literature. The analysis shows that the existing schemes are failing to satisfy desirable attributes. We have also analyzed the security of recently proposed Yeh’s multi-server authentication scheme. We have demonstrated the vulnerability of Yeh’s scheme to off-line password guessing attack, insider attack and user impersonation attack. Moreover, we have proposed an efficient multi-server authentication scheme which does not require all servers to be trusted. Moreover, central authority no longer required in mutual authentication and smart card need not to be stored multiple secret keys in the propose scheme. We have proved the correctness of mutual authentication of our scheme using BAN logic. Through the formal and informal security analysis, we have shown that our scheme is secure against various known attacks including the attacks found in Yeh’s scheme and other related schemes. In addition, the proposed scheme is comparable in terms of the communication and computational overheads with related multi-server authentication schemes.

References

1. Mishra, D. (2015). On the security flaws in ID-based password authentication schemes for telecare medical information systems. *Journal of Medical Systems*, *39*(1), 1–16.
2. Mishra, D., & Mukhopadhyay, S. (2014). Cryptanalysis of Yang et al.’s digital rights management authentication scheme based on smart card. *Recent Trends in Computer Networks and Distributed Systems Security*, *420*, 288–297.
3. Ojanperä, T., & Mononen, R. (2002). Security and authentication in the mobile world. *Wireless Personal Communications*, *22*(2), 229–235.
4. He, D., Zhang, Y., & Chen, J. (2014). Cryptanalysis and improvement of an anonymous authentication protocol for wireless access networks. *Wireless Personal Communications*, *74*(2), 229–243.
5. He, D., & Zeadally, S. (2015). Authentication protocol for an ambient assisted living system. *IEEE Communications Magazine*, *53*(1), 71–77.
6. Mishra, D., Chaturvedi, A., & Mukhopadhyay, S. (2015). An improved biometric-based remote user authentication scheme for connected healthcare. *International Journal of Ad Hoc and Ubiquitous Computing*, *18*(1–2), 75–84.
7. He, D., Kumar, N., & Chilamkurti, N. (2015). A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks. *Information Sciences*. doi:10.1016/j.ins.2015.02.010.
8. Shen, J., Tan, H., Wang, J., Wang, J., & Lee, S. (2015). A novel routing protocol providing good transmission reliability in underwater sensor networks. *Journal of Internet Technology*, *16*(1), 171–178.
9. Chaturvedi, A., Mishra, D., & Mukhopadhyay, S. (2013). Improved biometric-based three-factor remote user authentication scheme with key agreement using smart card. In *Information systems security* (pp. 63–77). Springer.

10. Moon, J. S., Park, J. H., Lee, D. G., & Lee, I.-Y. (2010). Authentication and ID-based key management protocol in pervasive environment. *Wireless Personal Communications*, *55*(1), 91–103.
11. Guo, P., Wang, J., Geng, X. H., Kim, C. S., & Kim, J.-U. (2014). A variable threshold-value authentication architecture for wireless mesh networks. *Journal of Internet Technology*, *15*(6), 929–936.
12. 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.
13. Li, L.-H., Lin, L.-C., & Hwang, M.-S. (2001). A remote password authentication scheme for multiserver architecture using neural networks. *IEEE Transactions on Neural Networks*, *12*(6), 1498–1504.
14. 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.
15. Cao, X., & Zhong, S. (2006). Breaking a remote user authentication scheme for multi-server architecture. *IEEE Communications Letters*, *10*(8), 580–581.
16. Juang, W.-S. (2004). Efficient multi-server password authenticated key agreement using smart cards. *IEEE Transactions on Consumer Electronics*, *50*(1), 251–255.
17. Chang, C.-C., & Lee, J.-S. (2004). An efficient and secure multi-server password authentication scheme using smart cards. In *2004 international conference on cyberworlds, IEEE* (pp. 417–422).
18. 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.
19. Chen, Y., Huang, C.-H., & Chou, J.-S. (2008). Comments on two multi-server authentication protocols. *IACR Cryptology ePrint Archive*, *2008*, 544.
20. Tsaur, W.-J., Li, J.-H., & Lee, W.-B. (2012). An efficient and secure multi-server authentication scheme with key agreement. *Journal of Systems and Software*, *85*(4), 876–882.
21. Chou, J.-S., Chen, Y., Huang, C.-H., & Huang, Y.-S. (2012). Comments on four multi-server authentication protocols using smart card. *IACR Cryptology ePrint Archive*, *2012*, 406.
22. 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.
23. Chen, T.-Y., Hwang, M.-S., Lee, C.-C., & Jan, J.-K. (2009). Cryptanalysis of a secure dynamic id based remote user authentication scheme for multi-server environment. In *2009 fourth international conference on innovative computing, information and control (ICICIC), IEEE* (pp. 725–728).
24. 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.
25. 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.
26. Truong, T.-T., Tran, M.-T., & Duong, A.-D. (2013). Robust secure dynamic id based remote user authentication scheme for multi-server environment. In *Computational science and its applications–ICCSA 2013* (pp. 502–515). Springer.
27. 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.
28. 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.
29. He, D., & Wang, D. (2015). Robust biometrics-based authentication scheme for multiserver environment. *IEEE System Journal, 9*(3), 816–823. doi:10.1109/JSYST.2014.2301517.
30. Wang, B., & Ma, M. (2013). A smart card based efficient and secured multi-server authentication scheme. *Wireless Personal Communications*, *68*(2), 361–378.
31. He, D., & Wu, S. (2013). Security flaws in a smart card based authentication scheme for multi-server environment. *Wireless Personal Communications*, *70*(1), 323–329.
32. Pippal, R. S., Jaidhar, C., & Tapaswi, S. (2013). Robust smart card authentication scheme for multi-server architecture. *Wireless Personal Communications*, *72*(1), 729–745.
33. He, D., Chen, J., Shi, W., & Khan, M. K. (2013). On the security of an authentication scheme for multi-server architecture. *International Journal of Electronic Security and Digital Forensics*, *5*(3), 288–296.
34. Yeh, K.-H. (2014). A provably secure multi-server based authentication scheme. *Wireless Personal Communications*, *79*(3), 1621–1634.
35. Burrows, M., Abadi, M., & Needham, R. M. (1989). A logic of authentication. *Proceedings of the Royal Society of London A: Mathematical and Physical Sciences*, *426*(1871), 233–271.
36. Syverson, P., & Cervesato, I. (2001). The logic of authentication protocols. In *Foundations of security analysis and design* (pp. 63–137). Springer.

37. Boyd, C., & Mao, W. (1994). On a limitation of ban logic. In *Advances in CryptologyEUROCRYPT93* (pp. 240–247). Springer.
38. Bellare, M., Canetti, R., & Krawczyk, H. (1996). Keying hash functions for message authentication. In *Advances in cryptology (CRYPTO'96)* (pp. 1–15). Springer.
39. Bellare, M., & Rogaway, P. (1997). Collision-resistant hashing: Towards making uowhfs practical. In *Advances in cryptology (CRYPTO'97)* (pp. 470–484). Springer.
40. Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of Computation*, *48*(177), 203–209.
41. Miller, V. S. (1986). Use of elliptic curves in cryptography. In *Advances in CryptologyCRYPTO85 proceedings* (pp. 417–426). Springer.
42. Boyd, C., & Mathuria, A. (2003). *Protocols for authentication and key establishment*. Berlin: Springer.
43. Eisenbarth, T., Kasper, T., Moradi, A., Paar, C., Salmasizadeh, M., & Shalmani, M. T. M. (2008). On the power of power analysis in the real world: A complete break of the keeloq code hopping scheme. In *Advances in cryptology-CRYPTO 2008* (pp. 203–220). Springer.
44. Kocher, P., Jaffe, J., & Jun, B. (1999). Differential power analysis. In *Advances in cryptology-CRYPTO'99* (pp. 388–397). Springer.
45. 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.
46. Dolev, D., & Yao, A. C. (1983). On the security of public key protocols. *IEEE Transactions on Information Theory*, *29*(2), 198–208.
47. Aumasson, J. P., Henzen, L., Meier, W., & Plasencia, M. N. (2010). Quark: A lightweight hash. In *Proceedings of workshop on cryptographic hardware and embedded systems (CHES 2010), lecture notes in computer science* (Vol. 6225, pp. 1–15). Springer.
48. Das, A. K., Massand, A., & Patil, S. (2013). A novel proxy signature scheme based on user hierarchical access control policy. *Journal of King Saud University: Computer and Information Sciences*, *25*(2), 219–228.
49. Abdalla, M., & Pointcheval, D. (2005). Interactive diffie–hellman assumptions with applications to password-based authentication. In *Financial cryptography and data security* (pp. 341–356). Springer.
50. Islam, S. H. (2014). Provably secure dynamic identity-based three-factor password authentication scheme using extended chaotic maps. *Nonlinear Dynamics*, *78*(3), 2261–2276.
51. Standard, S. H. FIPS PUB 180-1, National Institute of Standards and Technology (NIST), US Department of Commerce, April 1995. Accessed November 2010.

Dheerendra Mishra completed his Bachelor of Science and Master of Science degrees from Jiwaji University, India in 2003 and 2005, respectively. He received Ph.D. from the Indian Institute of Technology, Kharagpur, India, in 2014. Currently, he is working as an Assistant Professor in the Department of Mathematics, the LNM Institute of Information Technology, Jaipur, India. His research interests include digital rights management system, access control in cloud, cryptographic protocols.