Received: 19 June 2019 | Revised: 17 August 2019 | Accepted: 28 August 2019

DOI: 10.1002/spy2.92

ORIGINAL ARTICLE

A secure and robust elliptic curve cryptography-based mutual authentication scheme for session initiation protocol

Mahdi Nikooghadam | **Haleh Amintoosi**

Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

Correspondence
Haleh Amintoosi, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.
Email: amintoosi@um.ac.ir

Abstract

Session initiation protocol (SIP) is known as multimedia communication protocol based on IP, which is leveraged to provide signaling as well as instant messaging services. Since SIP services are widely used by Internet users, an important challenge is to supply mutual authentication between the SIP server and the user. Recently, Qui et al have presented an authentication and key agreement protocol for SIP and mentioned that their protocol is efficient and secure. In this article, we demonstrate that the protocol proposed by Qui et al is not able to provide mutual authentication and is prone to various attacks including Denning-Sacco and denial of service attacks. We then propose a secure and efficient two-factor authentication and key agreement protocol for SIP using elliptic curve cryptography (ECC). We analyze the security of the proposed scheme and show that it is able to satisfy various security features and resist different types of attacks. We also compare the computation and communication costs of the proposed scheme with other related authentication schemes and demonstrate that the proposed scheme outperforms other known ECC-based methods in achieving low computation and communication costs as well as resisting against all known attacks.

KEYWORDS

authentication, cryptanalysis, elliptic curve cryptography, key agreement, session initiation protocol, voice over IP

1 | INTRODUCTION

Nowadays, with the advent of real-time applications such as instant messaging and voice/video calls, voice over IP (VoIP) technology has emerged, which is able to deliver voice communications over IP networks. In order to initiate, preserve, and stop multimedia sessions among those participating in a session, VoIP services require session initiation protocol (SIP), a client/server text based signaling protocol which was first developed by IETF on 1999.[1] SIP has been used in many applications such as file transfer, video conferences, voice/video distribution, and online games.[2] In order to use the SIP service, the user and the server authenticate each other mutually to prevent unauthorized user to utilize the multimedia services and at the same time, establish a session key for further secure communications. Once the authentication process and key establishment are done successfully, a secure channel is established between the two parties using the session key and they are now able to exchange their multimedia data in a secure manner by executing SIP. Designing a secure SIP authentication and key agreement scheme is a challenging and important issue for SIP. Therefore, different SIP authentication and key agreement schemes have been developed.[3–7]

Common authentication schemes normally rely on a single factor such as passwords.[8] However, encountering various attacks including password guessing and impersonation attacks[9] has lead to considering a second factor based such as smart card, which is hard to forge or copy.[10,11] In 2018, Qui et al[12] analyzed the authentication and key agreement protocol presented for SIP by Kumari et al[6] and pointed out that it is not able to provide preverification and perfect forward secrecy. To address the limitations of Kumari et al's work, they proposed an improved scheme that is aimed at maintaining the benefits of the original scheme while providing security features. However, in this article, we show that Qui et al's scheme has multiple drawbacks and is not able to provide mutual authentication. Moreover, we propose an efficient and secure elliptic curve cryptography (ECC)-based authentication and key agreement scheme for SIP that is robust against known security attacks with lower computation and communication complexity, compared to related work.

Our contributions are as follows:

- We carry out cryptanalysis of Qui et al's scheme[12] and show it is not able to provide mutual authentication and is prone to Denning-Sacco attack and denial of service attack.
- We propose a novel secure authentication and key agreement protocol based on ECC that addresses the security flaws of Qui et al's scheme[12] and is able to provide mutual authentication and user anonymity. We also demonstrate that the proposed protocol is robust against various attacks including replay attack, Dennig-sacco attack, insider attack, user/server impersonation attack, and password guessing attack.
- We formally analyze the security of our protocol using Scyther tool[13] and show the correctness of the approach. We also analyze the performance of the proposed scheme and prove that our scheme is able to satisfy various security features.
- We also perform a comparative analysis between our proposed method and other related work in terms of computation and communication complexity and show that the proposed scheme incurs minimum computation and communication complexity, compared to most other ECC-based authentication schemes including.[12]

The structure of the paper is as follows. In Section 2, we present an overview on the related works on SIP authentication. Section 3 introduces basic concepts used throughout the paper. Section 4 illustrates Qui et al's scheme[12] and describes its security weaknesses. In Section 6, we propose our protocol with details. Section 7 describes the informal as well as formal security analysis of the proposed scheme using the Scyther tool. In Section 8, we analyze and compare the performance of our proposed scheme with other related works. Section 9 concludes the paper.

2 | RELATED WORK

Many works have focused on presenting secure authentication and key agreement schemes for SIP to date. The fist recognized work was done by Franks et al[14] in 1999 from HTTP digest authentication. However, Yang et al[9] demonstrated that Franks et al[15] are prone to off-line password guessing attack, server-spoofing attack and then, proposed an improved authentication scheme for SIP. In 2005, Durlanik et al[16] presented an efficient and secure approach for SIP authentication using ECC, which is being used in most authentication scheme due to its efficiency, the difficulty of discrete logarithm problem, and having keys with shorter length. In 2010, the work presented by Yoon et al[17] demonstrated that Durlanik's authentication scheme is prone to a series of attacks including off-line password guessing and Denning-Sacco. Subsequently, they proposed an ECC-based secure and efficient SIP authentication scheme whose aim is to utilize the key block size, speed, and security together. The work presented by Arshad and Ikram[15] in 2013 proved that Tsai's[18] lightweight authentication scheme fails to resist the stolen-verifier attack and password guessing attack. Moreover, it is unsuccessful in providing known-key secrecy and perfect forward secrecy. By "perfect forward secrecy," we refer to the feature in typical key agreement protocols that assures that the session keys will not be compromised even if any longterm such as the server's secret key is compromised.[19] To remedy Tsai scheme's issues, they proposed a mutual authentication scheme for SIP, which was based on elliptic curve discrete logarithm problem (ECDLP). However, Pu et al[20] claimed that the work by Arshad and Ikram's scheme[15] is prone to the password-guessing attack and instead, presented a new secure and efficient authentication and key agreement scheme for SIP, which is immune to this attack.

In 2014, Zhang et al[21] proposed a flexible password authenticated key agreement protocol for SIP with the aim of avoiding maintenance of a password or verification table. Their proposed method was shown to be secure against the server spoofing attack, replay attack, the stolen-verifier attack, the man-in-the-middle attack, and the Denning-Sacco attack. Later, Zhang et al[22] demonstrated that their previous scheme[21] is vulnerable against impersonation attack. To address this problem, authors proposed an improved protocol[21] which used smart card. In 2015, Jiang et al[23] also proved that the scheme by Zhang et al[21] is prone to the malicious insider impersonation attack. Further, they addressed the issue by proposing an efficient scheme that considers the coupling between the authenticators and the identity. However, Arshad and Nikooghadam[24] showed that Jiang et al's[23] scheme is vulnerable against user impersonation attack. In order to address the limitations of Zhang et al's scheme,[21] Irshad et al[25] developed an enhanced SIP authentication scheme using a single round-trip. However, Arshad et al[26] showed that the scheme proposed by Irshad et al[25] lacks user anonymity and mutual authentication and is not secure against user impersonation attack. They also proposed a performance-improved scheme.

Tu et al[27] in 2015 also proved that Zhang's scheme[21] is insecure against impersonation attack and developed an enhanced scheme to eliminate this drawback. However, Farash[10] showed that Tue's scheme[27] is still vulnerable against impersonation attack. Also, Farash and his colleagues[28] pointed out that the protocol by Zhang et al[21] is vulnerable to impersonation attack and password changing attack and proposed an improved authentication scheme to address these limitations.

In 2017, Chaudhry et al[4] showed that the scheme presented by Tu et al[27] is still vulnerable to server impersonation, replay and denial of service attacks, and lacks user anonymity. They also investigated Farash's enhancement[10] on Tu et al's scheme[27] and showed that it fails to provide user anonymity and is vulnerable to replay attack. Further, they proposed an anonymous authenticated key agreement scheme which was shown to be more secure and could be used in almost all lightweight environments.

The drawback of Farash's scheme[28] such as lack of a preauthentication in the smart card and off-line password guessing attack was also demonstrated by Lu et al,[29] who then developed an anonymous modified scheme with ECC to address Farash's scheme security weaknesses.

In 2017, a biometric-based authentication protocol for SIP networks was proposed by Zhang et al.[30] Thereafter, Irshad et al[31] pointed out that Zhang's scheme[30] is vulnerable against multiple attacks such as privileged insider attack, session specific temporary attack, and denial-of-service attack and further proposed a secure scheme addressing the flaws of Zhang et al scheme.[30]

Zhang et al[32] in 2015 proposed an authentication scheme for SIP and claimed their scheme could resist various attacks while maintaining efficiency. However, Lu et al[8] illustrated that their scheme is vulnerable to insider attacks and did not provide mutual authentication. They then proposed an improved secure mutual authentication scheme to overcome the security weaknesses in Zhang et al[32] scheme. Nikooghadam et al[33] pointed out that Chaudhry et al's[4] work is prone to password guessing attack and proposed an scheme to tackle this issue.

In 2018, Sureshkumar et al[34] showed that Lu et al's scheme[8] does not provide user anonymity and mutual authentication and fails to overcome user impersonation and server impersonation attack. Further, they presented an improved mutual authentication and key establishment protocol and showed that their scheme is secure against ID/password guessing attacks. Also, in 2018, Ravanbakhsh et al[19] claimed that the presented protocols by Chaudhry et al[35] and Nikooghadam et al[33] are not able to afford the perfect forward secrecy. They also proved that the presented protocol by Zhang et al[3] is prone to known session-specific temporary information attack and replay attack and cannot afford user anonymity. Then, they proposed a two-factor authentication and key agreement protocol which is able to resist against multiple active and passive attacks.

In 2019, Sourav et al[5] demonstrated the security flaws of Sureshkumar et al[34] and Zhang et al[32] schemes and then, proposed an enhancement over Sureshkumar et al's scheme to address its security flaws without increasing the computational cost. Also, Dhillon et al[7] in 2019 proposed a new biometric-based authentication scheme using ECC for SIP based VoIP communications that uses three users' personal biometric with the aim of providing strong identity check and enhanced security.

In 2015, Kumari et al[6] analyzed Farash's[10] work and pointed out that it is insecure regarding a series of attacks including user impersonation, password guessing attack, and session-specific temporary information leakage attack and fails to provide user anonymity. They further proposed an enhanced scheme to overcome Farash's scheme limitations. However, in 2018, Qui et al[12] analyzed Kumari's authentication and key agreement scheme and showed that it fails to provide pre-verification and perfect forward secrecy. They then proposed an improved scheme to address these security flaws. However, their proposed scheme still suffers from major security issues, as shown in this paper.

The above analysis demonstrates that most of the proposed protocols still have some security flaws and cannot guarantee secure communication. Therefore, designing a more efficient and secure authentication and key agreement protocol for SIP is still a challenging academic topic.

3 | BACKGROUND

3.1 | Session initiation protocol

VoIP application makes use of the SIP to initiate, establish, and stop multimedia sessions. First developed by IETF on 1999,[1] SIP is a signaling protocol which is being used in multimedia applications including video inferencing and multimedia distribution.[1]

To get SIP services, a client initiates registration process with the server, which includes receiving a message from client containing his secret information like his identity/user name and password using some secure channel. After registration, the client is allowed to login with the server via using secrets shared previously on public channel. Next, another SIP client is located by SIP session procedure in order to establish a session. The following messages are exchanged between the client and server during the login/authentication procedure:

- REQUEST: Client sends a connection request to server.
- CHALLENGE: Once the request is received, a challenge message is sent from server to client. The challenge message normally includes random nonce and realm (used to prompt the username and password) as well as verification information to verify the validity of server.
- RESPONSE: Upon receiving the challenge message, the client fist verifies the legitimacy of the server, and then, sends a response message to it.
- Once the response message is received at the server, it first verifies the legitimacy of the user and if so, a session key is shared between them. Otherwise the session is terminated.

3.2 | Elliptic curve cryptography

ECC is a public key cryptography approach which is based on elliptic curves. An elliptic curve E over F_P is the set of all solutions $(x, y) \in F_P * F_P$ defined by Equation (1), where p is a large prime number.[36]

$$y^2 = x^3 + ax + b,$$
$$\text{where} \quad a, b \in F_P \quad \text{and} \quad 4a^3 + 27b^2 \neq 0. \tag{1}$$

Two basic elliptic curve operations are known as point addition and point multiplication.[37] Point multiplication, defined as Equation (2) and referred to as scalar multiplication, is computed using a series of addition and multiplication.[37]

$$kP = P + P + P + \cdots + P \ (k \ \text{ times}). \tag{2}$$

Assume that P, $Q \in E(F_P)$ such that $Q = nP$. Then, determining n given P and Q is difficult. This problem is called the ECDLP.[36] The hardness of the ECDLP enables several cryptographic schemes based on elliptic curves.

4 | REVIEW OF QUI ET AL'S SCHEME

In this section, the registration and authentication phases of Qui et al's scheme[12] is reviewed and its security flaws are explained. Qui et al's scheme[12] has four phases: initialization, registration, login and authentication, and password update phases. Table 1 depicts the notations used in this scheme.

4.1 | Initialization phase

In this phase, the server performs a set of initializations such as choosing a random number $k \in Z_p^*$ as the server's private key and computing $G = kP$ as the public key of S.

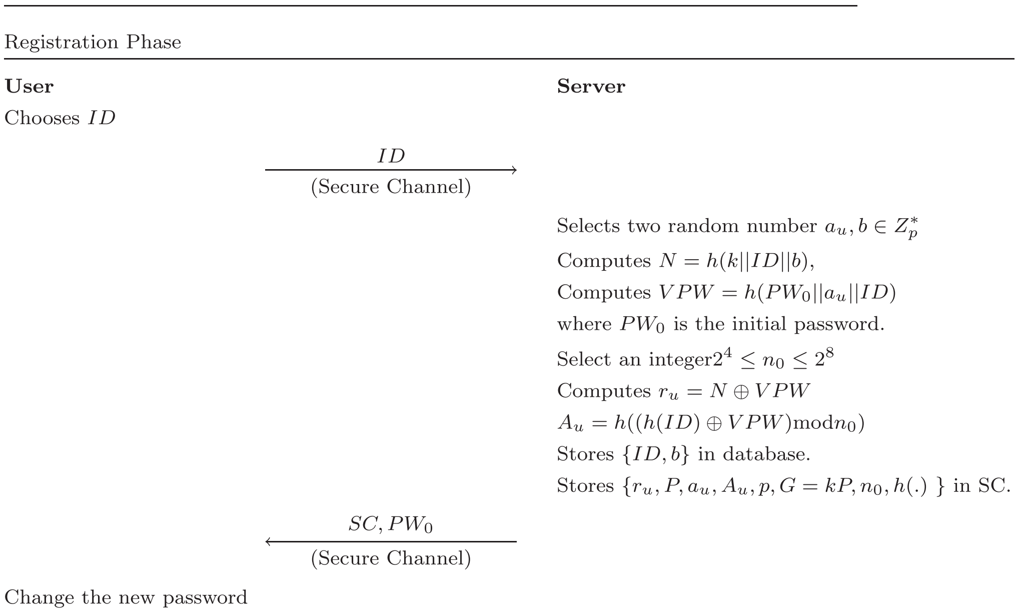
4.2 | Registration phase

The following steps are done between the server and the user. The result will be a smart card issued by the server to the user.

- Step 1. The user U selects an identity ID.
- Step 2. $U \Rightarrow S$: $\{ID\}$.
- Step 3. Once the registration message from U is received, S selects two random numbers $a_u, b \in Z_p^*$ and computes $N = h(k \| ID \| b)$ and $V\,PW = h(PW_0 \| a_u \| ID)$ where PW_0 is the initial password. The server S then selects an integer $2^4 \leq n_0 \leq 2^8$ and computes $r_u = N \oplus V\,PW$ and $A_u = h((h(ID) \oplus V\,PW) \bmod n_0)$. Finally, $\{ID, b\}$ is stored by S in its database.
- Step 4. The smart card SC contains $\{r_u, P, a_u, A_u, p, G = kP, n_0, h(.)\}$ and $S \Rightarrow U$: $\{SC, PW_0\}$.

TABLE 1 Notations used in Qui et al's scheme[12]

Symbol	Description
S	Server
U	Patient/user
ID	Identity of U
PW	Password of U
c_u, a_u	Random numbers of U
k_s	Secret key of S
b, c_s	Random numbers of S
$\|\|$	The string concatenation operation
$\oplus$	Bitwise (XOR) operation
$\mathcal{A}$	Malicious adversary
$h(.)$	Collision free one-way hash function
$\rightarrow$	An insecure channel
$\Rightarrow$	A secure channel
sk	Session key between U and S



Registration Phase

User

Chooses ID

$\xrightarrow[\text{(Secure Channel)}]{ID}$

Server

Selects two random number $a_u, b \in Z_p^*$

Computes $N = h(k||ID||b)$,

Computes $VPW = h(PW_0||a_u||ID)$

where PW_0 is the initial password.

Select an integer$2^4 \leq n_0 \leq 2^8$

Computes $r_u = N \oplus VPW$

$A_u = h((h(ID) \oplus VPW)\text{mod}n_0)$

Stores $\{ID, b\}$ in database.

Stores $\{r_u, P, a_u, A_u, p, G = kP, n_0, h(.)\ \}$ in SC.

$\xleftarrow[\text{(Secure Channel)}]{SC, PW_0}$

Change the new password

FIGURE 1 Registration phase of Qui et al[12]

- Step 5. Once the user U receives the smart card SC from S, the user changes the initial password during the password update phase.

Figure 1 depicts the registration phase of Qui et al's scheme.[12]

4.3 | Login and mutual authentication phase

- Step 1. U inserts his smart card and enters his ID, PW.
- Step 2. SC computes $VPW = h(PW||a_u||ID)$ and $A_u{}' = h((h(ID) \oplus VPW) \bmod n_0)$. Then SC compares whether $A'_u \overset{?}{=} A_u$. If so, it can be inferred that ID and PW are valid, otherwise the session is terminated.
- Step 3. SC calculates $N = r_u \oplus VPW$, selects a random number $c_u \in Z_p^*$, and calculates $V = c_uP$, $W = c_uG, f_u = ID \oplus W_x$, and $z_u = h(ID||W_y||f_u||N)$.

Login and Authentication Phase

User		**Server**
Inserts his/her SC and input ID, PW		
Computes $VPW = h(PW \| \| a_u \| \| ID)$		
Computes $A_u^{'} = h((h(ID) \oplus VPW) \bmod n_0)$		
$A_u^{'} =^? A_u$		
Computes $N = r_u \oplus VPW$		
Generates: $c_u \in Z_p^*$		
Computes $V = c_u P, W = c_u G$		
$f_u = ID \oplus W_x$		
$z_u = h(ID \| \| W_y \| \| f_u \| \| N)$		
	$\xrightarrow{\{V, f_u, z_u\}}$	
		Computes $W^* = kV$
		$ID = f_u \oplus W_x^*$
		Checks $ID^{'} =^? ID$
		Computes $z_u^* = h(ID \| \| W_y^* \| \| f_u \| \| N)$
		Checks $z_u^* =^? z_u$
		Generates $c_s, t \in Z_p^*$, compute$V_s = c_s V$
		$sk = h(N \| \| W_x^* \| \| G \| \| V_s \| \| ID \| \| t)$
		$Auth_s = h(t \| \| sk \| \| N)$
	$\xleftarrow{\{c_s G, Auth_s, t\}}$	
$V_s^* = c_u(c_s G)$		
$sk^* = h(N \| \| W_x \| \| G \| \| V_s^* \| \| ID \| \| t)$		
$Auth_s^* = h(t \| \| sk \| \| N)$		
Checks $Auth_s^* =^? Auth_s$		
If valid, then S is authenticated.		
Computes $Auth_u = h(t + 1 \| \| sk^* \| \| N \| \| V_s^* \| \| ID)$		
	$\xrightarrow{\{Auth_u\}}$	
		Computes $Auth_u^* = h(t + 1 \| \| sk \| \| N \| \| V_s \| \| ID)$
		Checks $Auth_u^* =^? Auth_u$
		If valid then U is authenticated
		Session Key: $sk = sk^*$

FIGURE 2 Authentication phase of Qui et al[12]

- Step 4. $U \rightarrow S : \{V, f_u, z_u\}$.
- Step 5. Once $\{V,\ f_u,\ z_u\}$ is received, S computes $W* = kV$ and $ID = f_u \oplus W_x^*$ and checks whether $ID' =^? ID$. If not, the password is wrong. Otherwise, S calculates $z_u^* = h(ID \| W_y^* \| f_u \| N)$ and verifies $z_u^* =^? z_u$. If not, the session is terminated. Otherwise, S chooses a random number $c_s, t \in Z_p^*$ and calculates $V_s = c_s V$, $sk = h(N \| W_x^* \| G \| V_s \| ID \| t)$ and $Auth_s = h(t \| sk \| N)$.
- Step 6. $S \rightarrow U : \{c_s G, Auth_s, t\}$.
- Step 7. Once the message is received, U calculates $V_s^* = c_u(c_s G)$, $sk^* = h(N \| W_x \| G \| V_s^* \| ID \| t)$, and $Auth_s^* = h(t \| sk^* \| N)$, and verifies if $Auth_s^* =^? Auth_s$. If not, the session is terminated.Otherwise, S is authenticated by U. Next, U calculates $Auth_u = h(t + 1 \| sk^* \| N \| V_s^* \| ID)$ and sends it to S.
- Step 8. $U \rightarrow S : Auth_u$.
- Step 9. Once $Auth_u$ is received, S calculates $Auth_u^* = h(t + 1 \| sk \| N \| V_s \| ID)$ and verifies if $Auth_u^* =^? Auth_u$. If true, U is authenticated.
- Step 10. At last, both U and S agree on a shared session key $sk = sk*$.

Figure 2 demonstrates the login and mutual authentication phase of Qui et al's scheme.[12]

5 | CRYPTANALYSIS OF QUI ET AL'S SCHEME

In this section, we explain with detail that Qui et al's scheme[12] does not provide mutual authentication. Besides, we demonstrate that the session keys agreed between the user and the server is not identical, showing that the protocol does not work correctly. Last but not least, the protocol is vulnerable to Denning-Sacco attack and denial of service attack.

5.1 | Session key unequality

As mentioned at step 10 of the authentication phase, in the end, the user U and the server S agree on a common session key $sk = sk*$. Here, we show that sk and $sk*$ are not the same.

- Step 1. As shown in step 7 of Qui et al's[12] authentication phase, the session key $sk*$ created at user side is $sk^* = h(N\|W_x\|G\|V_s^*\|ID\|t)$ where $V_s^* = c_u(c_sG)$.
- Step 2. As mentioned at the initialisation phase, G has been set as $G = kP$. So, $V_s^* = c_u(c_sG) = c_uc_skP$.
- Step 3. On server side, as mentioned in step 5 of the authentication phase, the session key sk is calculated as $sk = h(N\|W_x^*\|G\|V_s\|ID\|t)$, where V_s has been set to $V_s = c_sV$.
- Step 4. However, V is computed by the user (at step 3) as $V = c_uP$ and sent to the server. So, $V_s = c_sV = c_sc_uP$.
- Step 5. This indicates that $V_s \neq V_s^*$ resulting in $sk \neq sk*$.

Accordingly, we infer that contrary to the authors' claim, the session keys sk and $sk*$ are not equal at user and server sides. This implies that the key agreement is not done correctly.

5.2 | Mutual authentication

In order to provide mutual authentication, the following steps are done in Qui et al's[12] proposed scheme:

- Step 1. To authenticate the server S, at step 5 of the authentication phase, the server computes $Auth_s = h(t\|sk\|N)$ and sends it to the user via the message. Once the message is received, the user U computes $Auth_s^* = h(t\|sk^*\|N)$, and verifies if $Auth_s^* \overset{?}{=} Auth_s$. If so, S is authenticated by U.
- Step 2. To authenticate the user, at step 7 of the authentication phase, the user computes $Auth_u = h(t+1\|sk^*\|N\|V_s^*\|ID)$ and sends it to S. once received, as mentioned in step 9, the server calculates $Auth_u^* = h(t+1\|sk\|N\|V_s\|ID)$ and verifies if $Auth_u^* \overset{?}{=} Auth_u$. If true, U is authenticated by the server.
- Step 3. We demonstrated in Section 5.1 that sk and $sk*$ are not equal. Since $Auth_s^* \overset{?}{=} Auth_s$ and $Auth_u^* \overset{?}{=} Auth_u$ depend on the $sk* \overset{?}{=} sk$, unequality of sk and $sk*$ prevents the user and the server to authenticate each other correctly.

To conclude, Qui et al's scheme does not provide mutual authentication.

5.3 | Denning-Sacco attack

This attack refers to getting access to a long term private key such as the server's password, through an obtained old session key.[7] In the following, we demonstrate in two scenarios that Qui et al's scheme is vulnerable to Denning-Sacco attack.

Scenario 1:

- Step 1. As mentioned in step 5 of the authentication phase, the session key sk is computed as $sk = h(N\|W_x^*\|G\|V_s\|ID\|t)$. If sk is compromised, the adversary gets access to N and ID.
- Step 2. Having ID, the adversary is able to search in the server's database and obtain b. Note that $\{ID, b\}$ has been stored in the server's database, as mentioned at step 3 of the registration phase and the database has not been considered secure.
- Step 3. Having ID, N, and b, the adversary can run the brute force attack and obtain the server's private key k as $N = h(k\|ID\|b)$ (as mentioned in step 3 of the registration phase).

Scenario 2:

- Step 1. If sk is compromised, the adversary gets access to N, W_x^*, G, V_s, ID, t.

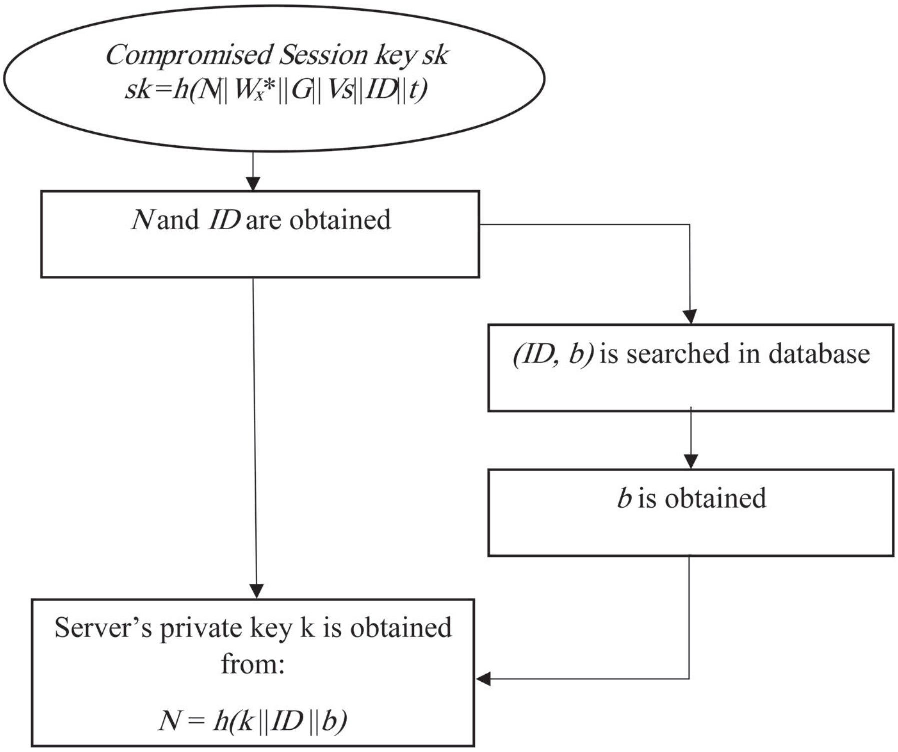

Compromised Session key sk
sk=h(N|| Wx*||G|| Vs||ID||t)
N and ID are obtained
(ID, b) is searched in database
b is obtained
Server's private key k is obtained from:
N = h(k||ID||b)

FIGURE 3 Attack diagram of Denning-Sacco attack on Qui et al's scheme, scenario 1

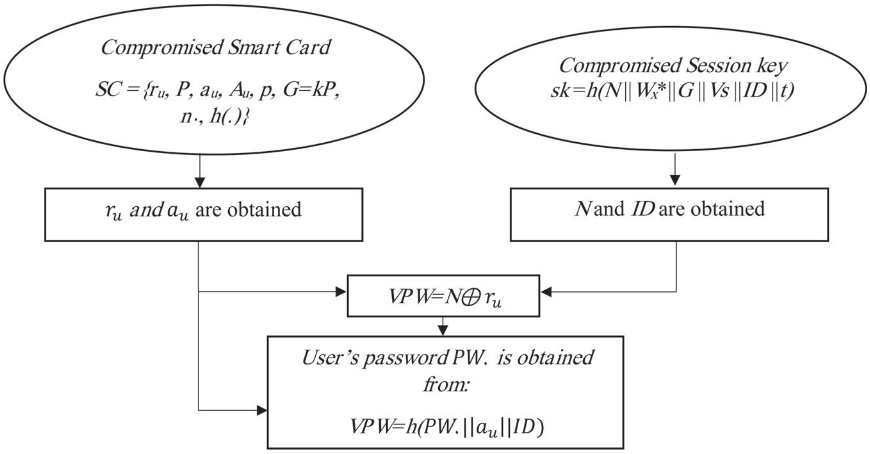


FIGURE 4 Attack diagram of Denning-Sacco attack on Qui et al's scheme, scenario 2

- Step 2. Also, if the adversary gets access to the smart card due to being lost or stolen, he gets access to its parameters including r_u and a_u. Having N from sk and r_u from the smart card, the adversary is able to compute $V\,PW$ using $r_u = N \oplus V\,PW$ due to the reversibility of XOR function.
- Step 3. As mentioned in step 3 of the registration phase, $V\,PW$ is computed as $V\,PW = h(PW_0 \| a_u \| ID)$. In this equation, all parameters are available except for PW_0. Since the password is short in terms of number of bits, the adversary is able to perform the brute force attack and guess the user's password PW_0.

To be brief, Qui et al's scheme is not able to resist Denning-Sacco attack. Figures 3 and 4 show the attack dagrams related to the above-mentioned scenarios of Dennig-sacco attack.

5.4 | Denial of service attack

As mentioned in step 4 of the authentication phase, the user sends the message $\{V, f_u, z_u\}$ to the server. Since no time stamp has been set to avoid the message replay, the adversary is able to send the message multiple times, causing the authentication process and specifically, the expression $V_s = c_s V$ (which contains a scalar multiplication with high computational complexity) to be repeatedly executed. This process leads to the service being denied by the server.

TABLE 2 Notations used in the proposed scheme

Symbol	Description
U_i	User i
S	The SIP server
ID_i	Identity of U_i
pw_i	Password of U_i
q_s	A high-entropy secret key of S
SC	The smart card
p	The base point of the elliptic curve
a_i, b_i, c_i, d_i, n_i	High entropy random numbers
$\|\|$	concatenation operation
$\oplus$	Bitwise (XOR) operation
SK	The shared one-time session key
T_1, T_2, T_3	The current time of user's system/server's system
$E_k(.)/D_k(.)$	The symmetric encryption/decryption with the key k
$h(.)$	A secure one-way hash function
ΔT	The maximum transmission delay

6 THE PROPOSED SCHEME

In this section, we describe our proposed secure and efficient ECC-based authentication and key agreement protocol for SIP. The novelty and strength of our scheme is as follows:

- We propose a novel secure and efficient authentication and key agreement scheme based on ECC for SIP, with the aim of providing various security requirements and resisting known security attacks while incurring very low computation/communication overhead.
- Our proposed scheme is resistant against almost all security threats including insider attack, known-session-specific temporary information attack, user impersonation attack, server impersonation attack, replay attack, offline password guessing attack, Denning-Sacco attack, and denial of service attack, as compared to recent related work including.[12] It is also able to provide mutual authentication, user anonymity, perfect forward secrecy, and known key secrecy. Excessive discussion will be presented in Section 7.
- The proposed authentication scheme is able to achieve very low computational complexity (ie, 30 ms), compared to some other ECC-based schemes[3,10,12,23,27,38,39] (between 44 and 66 ms). The details of performance analysis will be discussed in Section 8.
- The proposed authentication scheme is also able to achieve minimum communication complexity (ie, 1280 bits), compared to some other ECC-based schemes[3,10,12,23,27,39] (between 1344 and 1536 bits, except for Irshad et al[38] which is 1152 bits). The details of comparative analysis will be discussed in Section 8.

The protocol has three steps: registration, authentication and key agreement, and password update. Table 2 demonstrates the notations used in the proposed protocol.

6.1 Registration phase

The server and the user perform the following steps. At the end of this phase, a smart card is issued by server which is delivered to the user.

- Step 1. The user selects an identity ID_i, a password pw_i, and two random numbers a_i and b_i. Then, he calculates $bID_i = ID_i \oplus b_i$ and $mpw_i = (pw_i \oplus bID_i) \oplus pw_i$ and sends $\{bID_i, mpw_i, b_i, a_i\}$ to the server on a secure channel.

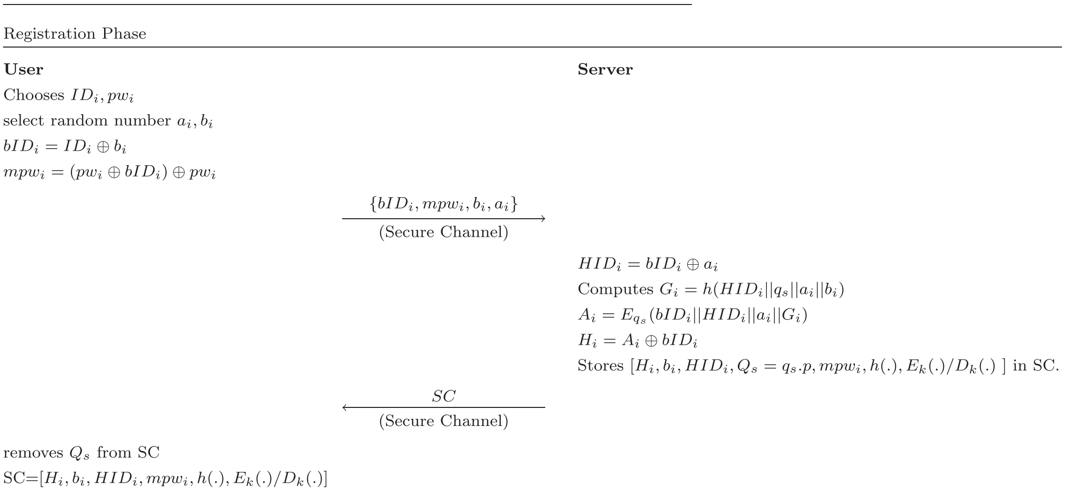

Registration Phase
User
Server
Chooses ID_i, pw_i
select random number a_i, b_i
$bID_i = ID_i \oplus b_i$
$mpw_i = (pw_i \oplus bID_i) \oplus pw_i$
$\{bID_i, mpw_i, b_i, a_i\}$
(Secure Channel)
$HID_i = bID_i \oplus a_i$
Computes $G_i = h(HID_i||q_s||a_i||b_i)$
$A_i = E_{q_s}(bID_i||HID_i||a_i||G_i)$
$H_i = A_i \oplus bID_i$
Stores $[H_i, b_i, HID_i, Q_s = q_s.p, mpw_i, h(.), E_k(.)/D_k(.)$] in SC.
SC
(Secure Channel)
removes Q_s from SC
SC=$[H_i, b_i, HID_i, mpw_i, h(.), E_k(.)/D_k(.)]$

FIGURE 5 Registration phase of the proposed scheme

- Step 2. Upon receiving the parameters, the server computes the following parameters:
  - $HID_i = bID_i \oplus a_i$
  - $G_i = h(HID_i \| q_s \| a_i \| b_i)$
  - $A_i = E_{q_s}(bID_i \| HID_i \| a_i \| G_i)$
  - $H_i = A_i \oplus bID_i$
- The server then stores (H_i, b_i, HID_i, $Q_s = q_s.p$, mpw_i, $h(.)$, $E_k(.)/D_k(.)$) in the smart card and sends it to the user through a secure channel.
- Step 3. Once the smart card is received, the user extracts Q_s, keeps it for further use, and then removes it from the smart card. At the end, the smart card contains $SC = [H_i, b_i, HID_i, mpw_i, h(.), E_k(.)/D_k(.)]$.

Figure 5 demonstrates the registration step of the proposed scheme.

6.2 | Authentication phase

- Step 1. The user inserts his smart card and enters his ID_i^* and pw_i^*. Then, the following calculations are performed by the smart card:

$$bID_i^* = ID_i^* \oplus b_i.$$

$$mpw_i^* = (pw_i^* \oplus bID_i^*) \oplus pw_i^*.$$

It then checks whether $mpw_i^* \overset{?}{=} mpw_i$. If so, it is verified that the card belongs to the user. Otherwise, the session is terminated.

- Step 2. The user then selects a time stamp T_1 and random numbers c_i, d_i, n_i. Then he computes C_i and D_i as two points on the elliptic curve as $C_i = c_i.p$ and $D_i = d_i.p$, respectively. Then, E_i is obtained by adding C_i and D_i as $E_i = D_i + C_i$. The user then computes $key_1 = c_i.Q_s = c_i.q_s.p$ and $A_i^* = H_i \oplus bID_i^*$. It then encrypts $(C_i \| D_i \| A_i^* \| T_1 \| n_i)$ with key_1 to obtain F_i, as $F_i = E_{key_1}(C_i \| D_i \| A_i^* \| T_1 \| n_i)$. At the end, $\{C_i, F_i, T_1, E_i\}$ is sent to the server.
- Step 3. Once the message is received at the server, the server first selects a time stamp T_2 and checks the freshness of the message by checking whether $|T_2 - T_1| \leq \Delta T$. If it does not hold, the session is terminated. Otherwise, the server creates $key_1' = C_i.q_s = c_i.p.q_s$. As mentioned above, $key_1 = c_i.Q_s = c_i.q_s.p$. On the other hand, $key_1' = C_i.q_s = c_i.p.q_s$. This means that $key_1' = key_1$. In order to authenticate the received message, the following steps are done: (a) At first, the server decrypts F_i with key_1' as $D_{key_1'}(F_i) = \{C_i^*, D_i^*, A_i^*, T_1^*, n_i^*\}$ and obtains parameters $C_i^*, D_i^*, A_i^*, T_1^*, n_i^*$. (b) Then, it adds two points D_i^* and C_i^* to obtain E_i^* as $E_i^* = D_i^* + C_i^*$ and checks whether $E_i^* \overset{?}{=} E_i$. If so, the authentication is successful and the server assures that the message has been sent from the user.

- Step 4. Since A_i was encrypted with the server's secret key q_s in the registration phase, the server is now able to decrypt A_i and obtain parameters $bID_i^*, HID_i^*, a_i^*, G_i^*$ as $DEC_{q_s}(A_i) = (bID_i^* \| HID_i^* \| a_i^* \| G_i^*)$. It then calculates $key_2^* = D_i^* + E_i^*$ and $na_i^* = n_i^* \oplus a_i^*$ and finally, the session key SK as $SK = (HID_i^* \| G_i^* \| key_1' \| key_2^*)$. Then, the concatenations of na_i^*, G_i^*, and T_2 are encrypted with key_2^* to form $Auth_s$ as $Auth_s = E_{key_2^*}(na_i^* \| G_i^* \| T_2)$. Also, $bID_i^* = HID_i^* \oplus a_i^*$ and $z_i = h(bID_i, A_i)$ are calculated. Finally, $\{z_i, T_2, Auth_s\}$ are sent to the user.
- Step 5. Upon receiving the message, the smart card first selects the time stamp T_3 and verifies the freshness of the message by checking whether $|T_3 - T_2| < \Delta T$. If not so, the session is terminated. Otherwise, key_2' is calculated as $key_2' = D_i + E_i$ and then, $Auth_s$ is decrypted by key_2' as $DEC_{key_2'}(Auth_s) = (na_i^* \| G_i^* \| T_2^*)$ to obtain na_i^*, G_i^*, and T_2^*. It then calculates the following parameters:

$$a_i^* = n_i \oplus na_i^*.$$

$$HID_i^* = bID_i \oplus a_i^*.$$

$$z_i^* = h(bID_i, A_i).$$

- Next, it checks whether z_i^* equals to z_i. If it holds, the server is authenticated for the user. In that case, the smart card computes $SK = (HID_i \| G_i \| key_1 \| key_2')$ and $M_i = h(SK \| n_i + 1 \| a_i^* + 1 \| key_2')$ and sends M_i to the server.
- Step 6. Once M_i is received, the server first generates the time stamp T_4 and verifies the freshness of the message by checking whether $|T_4 - T_3| < \Delta T$. If so, the server calculates $M_i^* = h(SK \| n_i + 1 \| a_i + 1 \| key_2)$ and checks whether $M_i^* \overset{?}{=} M_i$. If it holds, the smart card is authenticated for the server. So, mutual authentication is guaranteed. Figure 6 demonstrates the authentication process of the proposed scheme.

6.3 | Password update phase

In this phase, the user is enabled to change his password in a secure manner. The steps are as follows:

- Step 1. The user insets the smart card and enters his current identity and password as ID_i^* and pw_i^*. Then, bID_i^* is calculated as $bID_i^* = ID_i^* \oplus b_i$ and $mpw_i^* = (pw_i^* \oplus bID_i^*) \oplus pw_i^*$.
- Step 2. Having mpw_i in the smart card, the smart card checks whether $mpw_i^* = mpw_i$ or not. If so, it is proved that the smart card belongs to the user.
- Step 3. The smart card requests the user to enter his new password pw_i^{**}. Then, mpw_i^{**} is computed as:

$$mpw_i^{**} = (pw_i^{**} \oplus bID_i^*) \oplus pw_i^{**}.$$

- At the end, the value mpw_i is replaced with mpw_i^{**} in the smart card.

7 | SECURITY ANALYSIS

In this section, we first present an informal security analysis of the proposed scheme and prove that the proposed scheme is secure against the most common security attacks. Then, we formally prove the security and correctness of the proposed scheme using the Scyther tool.

7.1 | Informal security analysis

Anonymity. To preserve the anonymity of the user, his identity ID_i should not be obtained by the adversary. Moreover, if the adversary eavesdrops the exchanged messages or if he finds/steals the smart card and extracts its stored information, he should not be able to acquire the user's identity ID_i. As shown in Figures 5 and 6, ID_i has not been used or exchanged directly within the protocol. Instead, $bID_i = ID_i^* \oplus b_i$ (where b_i is a random number) is used in both registration and authentication phases. Note that bID_i is not exchanged on a public channel. So, even if the attacker obtains the smart card and gets access to b_i, he is not able to get access to the user's identity ID_i. Hence, anonymity of the user has been preserved.

Login and Authentication Phase

User **Server**

Inserts his/her SC and input ID_i^*, pw_i^*

$bID_i^* = ID_i^* \oplus b_i$

$mpw_i^* = (pw_i^* \oplus bID_i^*) \oplus pw_i^*$

$mpw_i^* =^? mpw_i$

Select Time Stamp T_1

Select Random Numbers c_i, d_i, n_i

$C_i = c_i.p$, $D_i = d_i.p$, $E_i = D_i + C_i$

Computes $key_1 = c_i.Q_s = c_i.q_s.p$

$A_i^* = H_i \oplus bID_i^*$

$F_i = E_{key_1}(C_i||D_i||A_i^*||T_1||n_i)$

$$\xrightarrow{\{C_i, F_i, T_1, E_i\}}$$

Select Time Stamp T_2

$|T_2 - T_1| < \varDelta T$

Computes $key_1' = C_i.q_s = c_i.p.q_s$

$DEC_{key_1'}(F_i) = (C_i^*||D_i^*||A_i^*||T_1^*||n_i^*)$

$E_i^* = D_i^* + C_i^*$

$E_i^* =^? E_i$

$T_1^* =^? T_1$

$DEC_{q_s}(A_i) = (bID_i^*||HID_i^*||a_i^*||G_i^*)$

$key_2^* = D_i^* + E_i^*$

$na_i^* = n_i^* \oplus a_i^*$

$SK = (HID_i^*||G_i^*||key_1'||key_2^*)$

$Auth_s = E_{key_2^*}(na_i^*||G_i^*||T_2)$

$bID_i^* = HID_i^* \oplus a_i^*$

Computes $z_i = h(bID_i||A_i)$

$$\xleftarrow{\{z_i, T_2, Auth_s\}}$$

Select Time Stamp T_3

$|T_3 - T_2| < \varDelta T$

Computes $key_2' = D_i + E_i$

$DEC_{key_2'}(Auth_s) = (na_i^*||G_i^*||T_2^*)$

$a_i^* = n_i \oplus na_i^*$

$HID_i^* = bID_i \oplus a_i^*$

Computes $z_i^* = h(bID_i||A_i)$

$z_i^* =^? z_i$

$T_2^* =^? T_2$

$SK = (HID_i||G_i||key_1||key_2')$

$M_i = h(SK||n_{i+1}||a_i^* + 1||key_2')$

$$\xrightarrow{M_i}$$

Select Time Stamp T_4

$|T_4 - T_3| < \varDelta T$

$M_i^* = h(SK||n_{i+1}||a_{i+1}||key_2)$

$M_i^* =^? M_i$

FIGURE 6 Authentication phase of the proposed scheme

Insider attack. As mentioned in the registration phase in Section 6.1, the user does not send his password directly to the server. Instead, mpw_i as $mpw_i = (pw_i \oplus bID_i) \oplus pw_i$ is sent in the registration phase, from which, pw_i cannot be obtained. Therefore, our proposed scheme is secure against insider attack.

Known-session-specific temporary information attack. As mentioned in Reference 19, resistance against known-session-specific temporary information attack implies that if session random numbers a_i, b_i, c_i, d_i, n_i are unexpectedly disclosed to the attacker, he should not be able to retrieve session key SK. As mentioned in the authentication phase, the session key $SK = (HID_i^* \| G_i^* \| key_1' \| key_2^*)$ includes HID_i^*, which is obtained from decrypting A_i with the server's secret key q_s as $DEC_{q_s}(A_i) = (bID_i^* \| HID_i^* \| a_i^* \| G_i^*)$. Since the attacker does not access the server's secret key q_s, he is not able to obtain the session key HID_i and accordingly, SK. So, our proposed protocol is robust against known-session-specific temporary information attack.

User impersonation attack. As its name implies, in this attack, the adversary aims to impersonate himself as a legal user to the server.[19] In order for the attacker to impersonate the user, he can send to the server his own parameters F_i', E_i', and M_i' instead of F_i, E_i, and M_i, respectively, on the public channel. In the following, we express why the adversary is not able to impersonate the user regarding these three parameters:

- If the adversary sends his own parameter F_i' instead of F_i, the server is not able to decrypt it in step 3 of the authentication phase, since F_i has been encrypted with key_1' which itself is dependent on the server's secret key q_s that is unreachable for the attacker. This means that F_i cannot be forged and hence, the adversary cannot impersonate the user through his own F_i'.
- As mentioned in step 3 of the authentication phase, once F_i is decrypted at the server, its parameters including D_i^* and C_i^* are obtained. Then, in order to authenticate the user, the server calculates E_i^* as $E_i^* = D_i^* + C_i^*$ and checks whether E_i^* equals to E_i received from the user. As mentioned above, F_i and hence, its parameters D_i^* and C_i^* cannot be forged. So, if the adversary sends his own parameter E_i' instead of E_i to the server, the comparison of E_i' against E_i^* fails and the adversary is not authenticated. So, the adversary is not able to impersonate the user through his own E_i'.
- As mentioned in step 6 of the authentication phase, once M_i is received from the user, the server calculates $M_i^* = h(SK \| n_i + 1 \| a_i + 1 \| key_2)$ and checks whether $M_i^* \overset{?}{=} M_i$. If so, the user is authenticated. If the adversary tends to send his own M_i' instead of M_i, the comparison of M_i' with M_i^* fails since M_i^* is dependant on parameters such as SK, $n_i + 1$ and $a_i + 1$ which are in the possess of the server and have not been exchanged elsewhere on public channel. So, the adversary is not able to impersonate the user through his own M_i'.

Server impersonation attack. This attack refers to the effort that the adversary makes in order to impersonate himself as a legal server to the user.[19] In order for the attacker to impersonate the server, he can send to the user his own parameters z_i' and $Auth_s'$ instead of z_i and $Auth_s$ respectively, on the public channel. In the following, we explain why the adversary is not able to impersonate the server.

- As mentioned in step 5 of the authentication phase, once z_i is received from the server, the user calculates $z_i^* = h(bID_i, A_i)$ and compares it with z_i. If equal, the server is authenticated. As can be seen, z_i^* depends on bID_i and A_i which are created by the user at the registration phase and are in the possess of the user. So, if the adversary tends to send his own z_i' instead of z_i, the comparison of z_i' with z_i^* fails and the session is terminated. So, the adversary is not able to impersonate the server through his own z_i'.
- As mentioned in step 5 of the authentication phase, once $Auth_s$ is received from the server, the user computes key_2' and decrypts $Auth_s$ using key_2'. If the adversary tends to send his own $Auth_s'$ instead of $Auth_s$, the user will not be able to decrypt $Auth_s'$ with key_2' and the session is terminated. So, the adversary is not able to impersonate the server through his own $Auth_s'$.

Replay attack. This attack refers to repeatedly sending an old message by the attacker.[6] Assume an attacker replays the old message as $\{C_i, F_i, T_1, E_i\}$ to the server. In our scheme, the server will find out that this message is old. At first, the server verifies $|T_2 - T_1| \leq \Delta T$, and if this condition is not true, the session terminates. Even if the attacker changes T_1 with current time T_1^{**} and sends $\{C_i, F_i, T_1^{**}, E_i\}$ to the server, the server is able to distinguish that the message is old. The server decrypts F_i with $key_1{}'$ as $DEC_{key_1'}(F_i) = (C_i^* \| D_i^* \| A_i^* \| T_1^* \| n_i^*)$ and compares T_1^* (obtained from decryption) with T_1^{**}. If not equal, the server identifies that the timestamp has been changed. The same stands for T_2 included in $\{z_i, T_2, Auth_s\}$ message in step 5 of the authentication process, where the user checks $|T_3 - T_2| \leq \Delta T$, and for T_4 in step 6, where the server checks $|T_4 - T_3| \leq \Delta T$. So, our proposed scheme is resistant against replay attacks.

Offline password guessing attack. As the name of the attack implies, if the attacker is able to acquire the exchanged messages, he should not able to obtain the user's password $pw_{i.}$[40] In our scheme, pw_i has not been exchanged anywhere in the protocol.

Instead, mpw_i as $mpw_i = (pw_i \oplus bID_i) \oplus pw_i$ is calculated from pw_i at the beginning of the registration phase, and exchanged in a secure manner. Also note that pw_i cannot be obtained from mpw_i due to using $\oplus$ in its calculation. So, our proposed scheme is robust against offline password guessing attack.

Known-key secrecy. This attack refers to obtaining the session key from the session keys belonging to previous sessions.[19] As mentioned above, the session key is equal to SK as $SK = (HID_i||G_i||key_1||key_2')$ where $key_1 = c_i.Q_s$ and $key_2' = D_i + E_i$ where $D_i = d_i.p$ and $E_i = D_i + C_i$. c_i and d_i are random numbers that are generated for each session and are not the same as the ones in previous sessions. So, even if the session key revealed, it is not possible for the attacker to compute the session keys belonging to other sessions.

Denning-Sacco attack. This attack refers to getting access to a long term private key such as the user's password or the session key, through an obtained old session key.[7] In our proposed protocol, the session key SK is $SK = (HID_i||G_i||key_1||key_2')$ in which, key_1 and key_2' include random numbers c_i and d_i that are selected at each session. So, if the attacker acquires old session key, he is not able to compute the server's secret key or other session keys. Moreover, since $mpw_i = (pw_i \oplus bID_i) \oplus pw_i$ is used instead of the user's password pw_i, even if the adversary gets access to the parameters exchanged on public channel, or the parameters within the smart card, he is not able to obtain pw_i. Besides, the server's secret key q_s has been only used once in calculating $G_i = h(HID_i||q_s||a_i||b_i)$. Since the attacker does not have access to a_i and b_i, he is not able to obtain q_s via brute force attack. This implies that the proposed scheme is resistant against Denning-Sacco attack.

Mutual authentication. In our proposed scheme, the server authenticates the user by verifying whether $E_i^* =^? E_i$ and $M_i^* =^? M_i$, respectively. On the other hand, the user authenticates the server by checking if $z_i^* =^? z_i$. Thus, our proposed scheme provides mutual authentication.

Denial of service attack. In our proposed scheme, timestamps have been used in all the steps which contain the scalar multiplication operation in order to check the freshness of the messages. Moreover, due to the utilization of random numbers in different steps of the authentication phase, the adversary is not able to run the denial of service attack, since the protocol does not allow sending repetitive messages. Subsequently, our proposed scheme is secure against denial of service attack.

Perfect forward secrecy. As mentioned before, perfect forward secrecy refers to the feature that ensures that the compromise of any longterm (eg, identifier, password, secret key, etc.) does not lead to the compromise of the session key.[19] In our proposed scheme, the session key $SK = (HID_i||G_i||key_1||key_2')$ includes key_2' which is computed as $key_2' = D_i + E_i$. On the other hand, D_i is computed as $D_i = d_i.p$ in which, d_i, is a random number. So, even by knowing the longterm, the adversary is not able to compute SK, due to its dependency to random numbers.

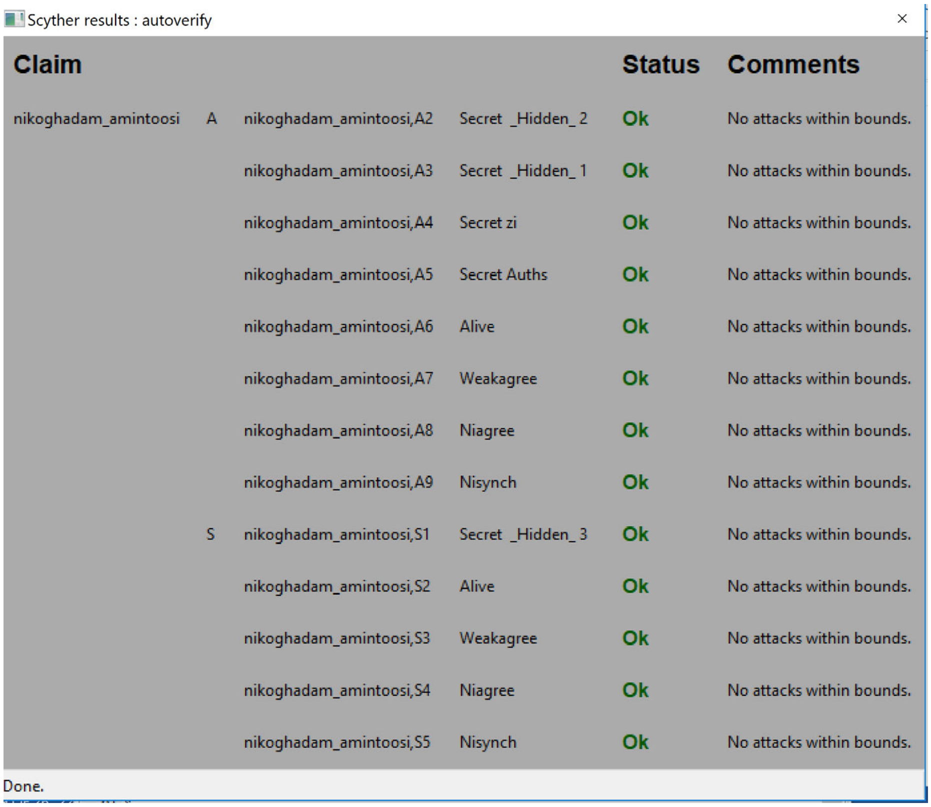
7.2 | Formal security analysis by Scyther tool

Scyther has been designed and extended as a tool with the aim of formally analyzing the security protocols and identifying their security requirements and vulnerabilities.[13] Scyther is based on the development model algorithm that provides the representation of traces, analyzes the protocol automatically, and examines its behavior against most of the potential attacks. Figure 7 depicts the analysis result of the proposed protocol via Scyther for 15 iterations. The term *Claim* is used to specify security requirements *Alive*, *Nisynch*, *weakagree*, and *secret*. The aim of using *Alive* is ensuring that some events have been executed by an intended communication party *R*. *Nisynch* means that all received messages are indeed sent by the sender and have been received by the receiver. *Claim*(*R*;*secret*;*rt*) means that *R* claims that *rt* must be unknown to an adversary. Finally, *weakagree* ensures that the protocol is secure against impersonation attack. As shown in Figure 7, the proposed protocol is able to satisfy all the above-mentioned security requirements. Scyther code has also been shown at the end of article.

8 | PERFORMANCE ANALYSIS

In this section, the results of performance analysis of our proposed method are presented. At first, the performance of the proposed scheme in regard to different security features is observed and compared to Farash,[10] Tu et al,[27] Zhang et al,[3] Jiang et al,[23] Qui et al,[12] Challa et al,[39] and Irshad et al.[38] Then, the computational complexity (ie, computation time in terms of milliseconds) is considered and calculated for our proposed scheme as well as the methods mentioned above. Finally, we compute the communication complexity (in terms of number of bits exchanged during the login and authentication phase) of the proposed scheme and compare it with the above mentioned related work.

The analysis of security features for our proposed scheme in comparison with the recent protocols has been presented in Table 3. As it can be observed, our suggested protocol is secure against all mentioned attacks and is able to provide security requirements such as anonymity and mutual authentication. Hence, our proposed scheme is able to provide a high level of security, compared to other existing authentication schemes.


Scyther results : autoverify

Claim				Status	Comments
nikoghadam_amintoosi	A	nikoghadam_amintoosi,A2	Secret _Hidden_ 2	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A3	Secret _Hidden_ 1	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A4	Secret zi	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A5	Secret Auths	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A6	Alive	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A7	Weakagree	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A8	Niagree	Ok	No attacks within bounds.
		nikoghadam_amintoosi,A9	Nisynch	Ok	No attacks within bounds.
	S	nikoghadam_amintoosi,S1	Secret _Hidden_ 3	Ok	No attacks within bounds.
		nikoghadam_amintoosi,S2	Alive	Ok	No attacks within bounds.
		nikoghadam_amintoosi,S3	Weakagree	Ok	No attacks within bounds.
		nikoghadam_amintoosi,S4	Niagree	Ok	No attacks within bounds.
		nikoghadam_amintoosi,S5	Nisynch	Ok	No attacks within bounds.

Done.

FIGURE 7 Security analysis of the proposed scheme using Scyther

TABLE 3 Comparison of security features

Security features	10	27	3	23	12	39	38	Ours
F_1	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
F_2	No	No	No	Yes	Yes	Yes	Yes	Yes
F_3	Yes	Yes	Yes	Yes	No	Yes	No	Yes
F_4	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
F_5	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
F_6	No	No	Yes	No	Yes	Yes	Yes	Yes
F_7	No	No	Yes	No	Yes	Yes	Yes	Yes
F_8	Yes	Yes	Yes	Yes	Yes	No	No	Yes
F_9	No	No	No	Yes	Yes	No	No	Yes
F_{10}	No	Yes	No	Yes	Yes	Yes	Yes	Yes
F_{11}	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
F_{12}	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes

F_1, provides mutual authentication; F_2, provides user anonymity and un-traceability; F_3, resists denial of service attack; F_4, resists privileged insider attack; F_5, resists Denning-Sacco attack; F_6, resists user impersonation attack; F_7, resists server impersonation attack; F_8, resists off/on-line password guessing attack; F_9, resists replay attack; F_{10}, resists session-specific temporary information attack; F_{11}, provides known-key secrecy; F_{12}, provides efficient password changing.

Table 4 shows the notations used to evaluate and compare the computational cost. In order to estimate the approximate execution timings, we use the experimental results presented in References 11,41, in which, the approximate execution timings of T_{hf}, T_{mu}, T_{ad}, and $T_{en/d}$ are 0.0004, 7.3529, 0.009, and 0.1303 ms, respectively. In our proposed scheme, three scalar multiplication operations, two symmetric encryption operations, one hash function operations and two point addition operations are required at the user side. Hence, the computational cost at the user side is $3T_{mu}+T_{hf}+2T_{en/d}+2T_{ad}$. Moreover, at the server side, one scalar multiplication operations, two hash function operations, four symmetric encryption operations, and two point addition operations are needed. So, the computational cost at the server side of the proposed scheme is $T_{mu}+2T_{hf}+4T_{en/d}+2T_{ad}$.

TABLE 4 Notations used in the computation cost analysis of the proposed scheme

Symbol	Description
T_{hf}	Time of performing a hash function operation
$T_{en/d}$	Time of performing symmetric encryption/decryption
T_{mu}	Time of performing the scalar multiplication operation of elliptic curve
T_{ad}	Time of performing a point addition operation of elliptic curve

TABLE 5 Computation cost comparison between the proposed protocol and related works

Scheme	User's computation	Server's computation	Total computation	Time (ms)
Farash[10]	$4T_{mu}+5T_{hf}+1T_{ad}$	$3T_{mu}+5T_{hf}$	$7T_{mu}+10T_{hf}+1T_{ad}$	51.4833
Tu et al[27]	$4T_{mu}+6T_{hf}$	$4T_{mu}+7T_{hf}$	$8T_{mu}+13T_{hf}$	58.8284
Zhang et al[3]	$3T_{mu}+5T_{hf}$	$3T_{mu}+6T_{hf}$	$6T_{mu}+11T_{hf}$	44.1218
Jiang Q et al[23]	$4T_{mu}+6T_{hf}+T_{ad}$	$5T_{mu}+6T_{hf}+T_{ad}$	$9T_{mu}+12T_{hf}+2T_{ad}$	66.1989
Qui et al[12]	$3T_{mu}+7T_{hf}$	$3T_{mu}+5T_{hf}$	$6T_{mu}+12T_{hf}$	44.1222
Irshad et al[38]	$4T_{mu}+1T_{en/d}+11T_{hf}$	$2T_{mu}+2\ T_{(}en/d)+5T_{hf}$	$6T_{mu}+3\ T_{(}en/d)+16T_{hf}$	44.5147
Challa et al[39]	$3T_{mu}+8T_{hf}$	$3T_{mu}+4T_{hf}$	$6T_{mu}+12T_{hf}$	44.1222
Ours	$3T_{mu}+T_{hf}+2T_{en/d}+2T_{ad}$	$T_{mu}+2T_{hf}+4T_{en/d}+2T_{ad}$	$4T_{mu}+3T_{hf}+6T_{en/d}+4T_{ad}$	30.2306

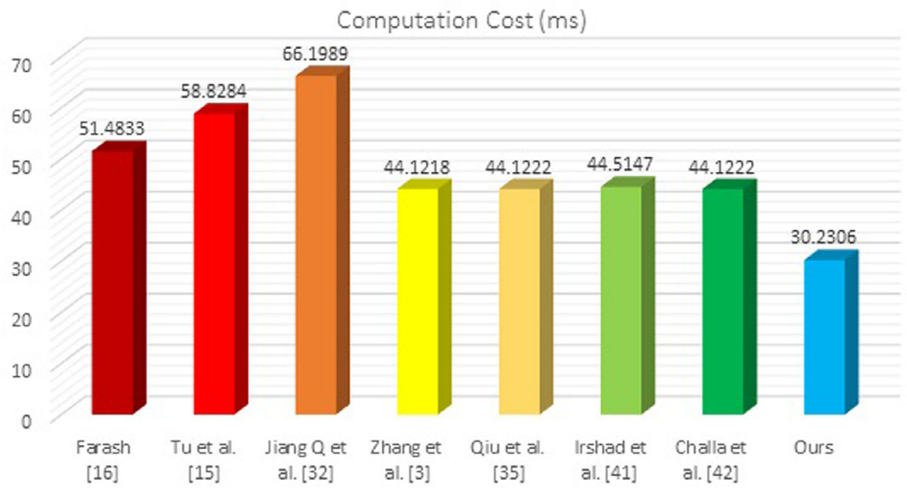


FIGURE 8 Comparison of execution time (in ms) between our proposed scheme and other schemes

Table 5 and Figure 8 show the results of comparing the computational time of our proposed method with Farash's work,[10] Tu et al,[27] Zhang et al,[3] Jiang et al,[23] Qui et al,[12] Challa et al,[39] and Irshad et al.[38] As shown in the table and figure, our proposed scheme outperforms other ECC-based schemes. Specifically, the total computation time of our scheme is 30.2306 ms, while, for example, it is 51.4833 ms for Farash,[10] 58.8284 ms for Tu et al,[27] 66.1989 ms for Jiang et al,[23] 44.1222 ms for Challa et al,[39] and 44.5147 ms for Irshad et al.[38] Also, as shown in Table 3, our scheme is able to withstand almost all security threats, compared to other ECC-based methods. In other words, our scheme is successful in achieving a delicate balance between the security and the performance while incurring minimum computational cost.

As a result, our proposed scheme outperforms other related schemes in terms of achieving all security requirements while showing the best computational performance.

Table 6 and Figure 9 demonstrate the comparison of communication cost (in terms of number of bits exchanged) of the proposed scheme with the schemes of Farash,[10] Tu et al,[27] Zhang et al,[3] Jiang et al,[23] Qui et al,[12] Challa et al,[39] and Irshad et al[38] in login and authentication phase. Based on References 19,40,41, communication cost of sending identity is considered to be 160 bits, timestamp is 32-bits, encryption/decryption operations is 128-bits, elliptic curve point multiplication is 320 bits, realm is 32 bits, random number and output hash function are 32 bits and 160 bits, respectively.

TABLE 6 Communication cost comparison between the proposed protocol and related works

Scheme	No. of messages	Fist message	Second message	Third message	Number of bits
Ours	3	C_i, F_i, T_1, E_i	$z_i, T_2, Auth_s$	M_i	1280
Qui et al[12]	3	V, f_u, z_u	$c_sG, Auth_s, t$	$Auth_u$	1440
Zhang et al[3]	3	ID_i, C_4, C_6	$Realm, C_7, Auth_s, r_4$	$Realm, Auth_u$	1376
Jiang et al[23]	3	$Username, V, W$	$Realm, Auth_s, S, r$	$Realm, Auth_u$	1536
Tuo et al[27]	3	$Username, V, W$	$Realm, Auth_s, C, r$	$Realm, Auth_u$	1376
Farash[10]	3	$Username, V, W$	$Realm, Auth_s, C, r$	$Realm, Auth_u$	1376
Irshad et al[38]	3	C, G	$realm, bP, Auth_s$	$realm, Auth_u$	1152
Challa et al[39]	2	DID_i, C_i, V_i, T_i	C_s, V_s, T_s	-	1344

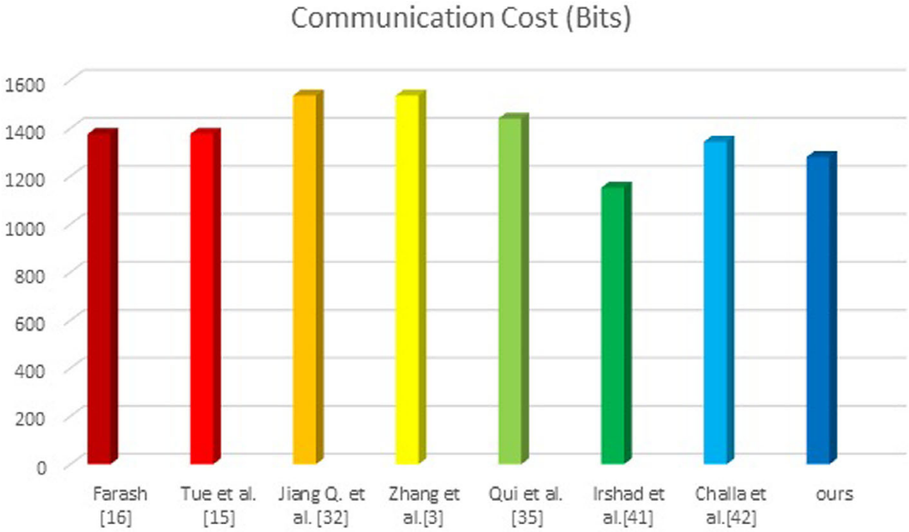


FIGURE 9 Comparison of communication cost (in bits) between our proposed scheme and other schemes

In proposed scheme, the message $\{C_i, F_i, T_1, E_i\}$ needs $(320+128+32+320) = 800$ bits, the message $\{z_i, T_2, Auth_s\}$ needs $(160+32+128) = 320$ bits, and the message $\{M_i\}$ needs 160 bits. Therefore, the proposed scheme requires $(800+320+160) = 1280$ bits for the communication cost of three messages transmitted between user and server. As we can see in Table 6, the proposed scheme has lower communication cost, compared to the schemes of Farash,[10] Tu et al,[27] Zhang et al,[3] Jiang et al,[23] Qui et al,[12] and Challa et al.[39]

9 | CONCLUSION

In this paper, we have first investigated the security weakness of Qui et al's scheme[12] and proved that it does not provide mutual authentication and is vulnerable against Denning-Sacco attack and denial of service attack. We then proposed an efficient and secure ECC-based two-factor authentication and key agreement scheme for SIP. We formally analyzed the security robustness of our proposed scheme and demonstrated that our scheme is able to satisfy all desirable security features and resists against different types of attacks. We also showed that our presented protocol requires a minimum computational and communication overhead compared to that for other ECC-based schemes. In future, we are going to redesign the protocol to have fewer scalar multiplication, leading to lower computational complexity. Moreover, we plan to present the lightweight version of the proposed protocol in our future research.

Scyther Code of the Proposed Protocol

```
usertype TimeStamp ;
const P;
```

```
hashfunction H1;
secret XOR: Function ;
secret ScalarMultiply : Function ;
secret jam : Function ;
secret JAM: Function ;
secret idi, pwi, ai, bi, ci, qs, di, ni, nai ;
macro bidi= XOR ( idi, bi ) ;
macro mpwi = XOR ( XOR (pwi, bidi ), pwi ) ;
macro Qs = ScalarMultiply ( qs, P) ;
macro hidi= XOR ( bidi, ai ) ;
macro Gi = H1( hidi, qs ) ;
macro Ai={bidi, hidi, ai, Gi}qs ;
macro Hi = XOR (Ai, bidi ) ;
protocol nikoghadam-amintoosi ( A, S )
{
r o l e A {
var Auths, zi ;
macro b i d i i= XOR ( idi, bi ) ;
macro mpwii = XOR ( XOR (pwi, bidi ), pwi ) ;
match(mpwii,mpwi ) ;
macro Ci = ScalarMultiply ( ci, P) ;
macro Di = ScalarMultiply ( di, P) ;
macro Ei = JAM(Ci, Di ) ;
macro key1 = ScalarMultiply ( ci, Qs ) ;
macro Ai = XOR (Hi, b i d i i ) ;
macro Fi={Ci, Di, Ai, ni }key1 ;
send_1 (A, S, ( Ci, Fi, Ei ) ) ;
recv_2 (S,A, ( zi, Auths ) ) ;
macro key22 = JAM(Ei, Di ) ;
macro Auths={nai, Gi}key22 ;
macro a i i= XOR ( ni, nai ) ;
macro hidi= XOR ( bidi, a i i ) ;
macro z i i= H1( bidi, a i i ) ;
match ( zii, zi ) ;
macro sk= H1( hidi, Gi, key1, key22 ) ;
macro Mi = H1( sk, jam( ni,1), jam( ai,1), key22 ) ;
send_3 (A, S, ( Mi ) ) ;
};
role S
{
recv_1 (A, S, ( Ci, Fi, Ei ) ) ;
macro key11 = ScalarMultiply (Ci, qs ) ;
match ( key11, key1 ) ;
macro Fi={Ci, Di, Ai, ni }key11 ;
macro Eii = JAM(Ci, Di ) ;
match( Eii, Ei ) ;
macro Ai={bidi, hidi, ai, Gi}qs ;
macro key2 = JAM(Ei, Di ) ;
macro nai= XOR ( ni, ai ) ;
macro sk= H1( hidi, Gi, key11, key2 ) ;
macro Auths={nai, Gi}key2 ;
macro bidi= XOR ( hidi, ai ) ;
macro zi= H1( bidi, Ai ) ;
send_2 (S,A, ( zi, Auths ) ) ;
recv_3 (A, S, ( Mi ) ) ;
macro Mii = H1( sk, jam( ni,1), jam( ai,1), key22 ) ;
match(Mi, Mii ) ;
};
};
```

CONFLICT OF INTEREST

The authors declare no potential conflict of interest.

ORCID

Haleh Amintoosi https://orcid.org/0000-0002-1447-8086

REFERENCES

1. Franks J, Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A, Stewart L, *HTTP Authentication: Basic and Digest Access Authentication*, (1999)
2. Tam K, Goh H. *Session initiation protocol*. *2002 IEEE International Conference on Industrial Technology, 2002. IEEE ICIT'02*. Vol 2. IEEE; 2002:1310-1314.
3. Zhang L, Tang S, Zhu S. An energy efficient authenticated key agreement protocol for SIP-based green VoIP networks. *J Netw Comput Appl*. 2016;59:126-133.
4. Chaudhry SA, Naqvi H, Sher M, Farash MS, Hassan MU. An improved and provably secure privacy preserving authentication protocol for sip. *Peer-to-Peer Netw Appl*. 2017;10(1):1-15.
5. Sourav S, Odelu V, Prasath R. *Enhanced session initiation protocols for emergency healthcare applications*. In: Thampi S, Madria S, Wang G, Rawat D, Alcaraz Calero J, eds. *Security in Computing and Communications. SSCC 2018. Communications in Computer and Information Science*. Vol 969; 2019.
6. Kumari S, Chaudhry SA, Wu F, Li X, Farash MS, Khan MK. An improved smart card based authentication scheme for session initiation protocol. *Peer-to-Peer Netw Appl*. 2015.
7. Dhillon PK, Kalra S. Secure and efficient ECC based SIP authentication scheme for VoIP communications in internet of things. *Multimedia Tools Appl*. 2019;1-24.
8. Lu Y, Li L, Peng H, Yang Y. A secure and efficient mutual authentication scheme for session initiation protocol. *Peer-to-Peer Netw Appl*. 2016;9(2):449-459.
9. Yang CC, Wang RC, Liu WT. Secure authentication scheme for session initiation protocol. *Comput Secur*. 2005;24(5):381-386.
10. Farash MS. Security analysis and enhancements of an improved authentication for session initiation protocol with provable security. *Peer-to-Peer Netw Appl.* vo. 9, no. 1, pp: 82–91, (2016)
11. Amin R, Islam SH, Biswas GP, Khan MK, Obaidat MS. Design and analysis of an enhanced patient-server mutual authentication protocol for telecare medical information system. *J Med Syst*. 2015;39(11).
12. Qiu S, Xu G, Ahmad H, Guo Y. An enhanced password authentication scheme for session initiation protocol with perfect forward secrecy. *PLOS One*. 2018;13(3):e0194072.
13. Cremers C. Scyther, Semantics and Verification of Security Protocols [Ph.D. dissertation]. Eindhoven University of Technology; 2006.
14. Frank J, Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A. *HTTP authentication: basic and digest access authentication. IETF RFC.2617*; 1999.
15. Arshad R, Ikram N. Elliptic curve cryptography based mutual authentication scheme for session initiation protocol. *Multimed Tools Appl*. 2013;6(2):165-178.
16. Durlanik A, Sogukpinar I. SIP authentication scheme using ECDH. *World Enformatika Society Transactions on Engineering Computing and Technology*. 2005;8:350–353.
17. Yoon EJ, Yoo KY, Kim C, Hong YS, Jo M, Chen HH. A secure and efficient SIP authentication scheme for converged VoIP networks. *Comput Commun*. 2010;33(14):1674-1681.
18. Tsai JL. Efficient nonce-based authentication scheme for session initiation protocol. *IJ Netw Secur*. 2009;9(1):12-16.

19. Ravanbakhsh N, Mohammadi M, Nikooghadam M. Perfect forward secrecy in VoIP networks through design a lightweight and secure authenticated communication scheme. *Multimedia Tools Appl*. 2018;78:11129-11153.
20. Pu Q, Wang J, Wu S. Secure SIP authentication scheme supporting lawful interception. *Secur Commun Netw*. 2013;6(3):340-350.
21. Zhang L, Tang S, Cai Z. Efficient and flexible password authenticated key agreement for voice over internet protocol session initiation protocol using smart card. *Int J Commun Syst*. 2014;7(11):2691-2702.
22. Zhang L, Tang S, Cai Z. Cryptanalysis and improvement of password-authenticated key agreement for session initiation protocol using smart cards. *Secur Commun Netw*. 2014;7(12):2405-2411.
23. Jiang Q, Ma J, Tian Y. Cryptanalysis of smart-card-based password authenticated key agreement protocol for session initiation protocol of Zhang et al. *Int J Commun Syst*. 2015;28(7):1340-1351.
24. Arshad H, Nikooghadam M. Security analysis and improvement of two authentication and key agreement schemes for session initiation protocol. *J Supercomput*. 2015;71(8):3163-3180.
25. Irshad A, Sher M, Rehman E, Ch SA, Hassan MU, Ghani A. A single round-trip sip authentication scheme for voice over internet protocol using smart card. *Multimed Tools Appl*. 2015;74(11):3967-3984.
26. Arshad H, Nikooghadam M. An efficient and secure authentication and key agreement scheme for session initiation protocol using ECC. *Multimedia Tools Appl*. 2016;75(1):181-197.
27. Tu H, Kumar N, Chilamkurti N, Rho S. An improved authentication protocol for session initiation protocol using smart card. *Peer-to-Peer Netw Appl*. 2015;8(5):903-910.
28. Farash MS, Attari MA. An anonymous and untraceable password-based authentication scheme for session initiation protocol using smart cards. *Int J Commun Syst*. 2016;29(13):1956-1967.
29. Lu Y, Li L, Peng H, Yang Y. An anonymous two-factor authenticated key agreement scheme for session initiation protocol using elliptic curve cryptography. *Multimed Tools Appl*. 2017;76(2):1801-1815.
30. Zhang L, Tang S, Zhu S. Privacy-preserving authenticated key agreement scheme basedon biometrics for session initiation protocol. *Wireless Netw*. 2017;23(6):1901-1916.
31. Irshad A, Chaudhry SA, Kumari S, Usman M, Mahmood K, Faisal MS. An improved lightweight multi-server authentication scheme. *Int J Commun Syst*. 2017;30(17):e3351.
32. Zhang Z, Qi Q, Kumar N, Chilamkurti N, Jeong HY. A secure authentication scheme with anonymity for session initiation protocol using elliptic curve cryptography. *Multimed Tools Appl*. 2015;74(10):3477-3488.
33. Nikooghadam M, Jahantigh R, Arshad H. A lightweight authentication and key agreement protocolpreserving user anonymity. *Multimed Tools Appl*. 2017;76(11):13401-13423.
34. Sureshkumar V, Amin R, Anitha R. A robust mutual authentication scheme for session initiation protocol with key establishment. *Peer-toPeer Netw Appl*. 2018;11(5):900-916.
35. Chaudhry SA, Farash MS, Naqvi H, Kumari S, Khan MK. An enhanced privacy preserving remote user authentication scheme with provable security. *Secur Commun Netw*. 2015;8(18):3782-3795.
36. Hankerson D, Menezes AJ, Vanstone S. *Guide to Elliptic Curve Cryptography*. Springer Science and Business Media; 2006.
37. Silverman JH. *The Arithmetic of Elliptic Curves*. Vol 106. Springer Science and Business Media; 2009.
38. Irshad A, Kumari S, Li X, Wu F, Chaudhry SA, Arshad H. An improved SIP authentication scheme based on server-oriented biometric verification. *Wirel Pers Commun*. 2017;97(2):2145-2166.
39. Challa S, Das AK, Kumari S, Odelu V, Wu F, Li X. Provably secure three-factor authentication and key agreement scheme for session initiation protocol. *Secur Commun Netw*. 2016;9(18):5412-5431.
40. Kumari S, Karuppiah M, Das AK, Li X, Wu F, Gupta V. Design of a secure anonymity preserving authentication scheme for session initiation protocol using elliptic curve cryptography. *J Ambient Intell Humaniz Comput*. 2018;9(3):643-653.
41. Xu L, Wu F. Cryptanalysis and improvement of a user authentication scheme preserving uniqueness and anonymity for connected health care. *J Med Syst*. 2015;39(10):10.

How to cite this article: Nikooghadam M, Amintoosi H. A secure and robust elliptic curve cryptography-based mutual authentication scheme for session initiation protocol. *Security and Privacy*. 2019;e92. https://doi.org/10.1002/spy2.92