Peer-to-Peer Netw. Appl. (2017) 10:92–105
DOI 10.1007/s12083-015-0409-0

An improved smart card based authentication scheme for session initiation protocol

Saru Kumari[1] · Shehzad Ashraf Chaudhry[2] · Fan Wu[3] · Xiong Li[4,5] · Mohammad Sabzinejad Farash[6] · Muhammad Khurram Khan[7]

Received: 14 March 2015 / Accepted: 21 August 2015 / Published online: 15 September 2015

Abstract Session initiation protocol (SIP) reformed the controlling routine of voice over Internet Protocol based communication over public channels. SIP is inherently insecure because of underlying open text architecture. A number of solutions are proposed to boost SIP security. Very recently Farash Peer to Peer Netw. Appl. 1–10, 2014 proposed an enhanced protocol to improve the security of Tu et al.'s protocol (Peer to Peer Netw. Appl. 1–8, 2014). Further, Farash claimed his protocol to be secure against all known attacks. However, in this paper we show that Farash's protocol is insecure against impersonation attack, password guessing attack, lacks user anonymity and is vulnerable to session-specific temporary information attack. Further, we have proposed an upgraded protocol to enhance the security. The security and performance analysis shows that the proposed protocol reduced one point multiplication as compared with Farash's protocol, while resisting all known attacks. We have proved the security of proposed protocol using automated tool ProVerif.

Keywords Authentication · Security · Anonymity and privacy · Impersonation attack · Provable security · ProVerif

✉ Saru Kumari
saryusiirohi@gmail.com

Shehzad Ashraf Chaudhry
shahzad@iiu.edu.pk

Fan Wu
conjurer1981@gmail.com

Xiong Li
lixiong84@gmail.com

Mohammad Sabzinejad Farash
sabzinejad@khu.ac.ir

Muhammad Khurram Khan
mkhurram@ksu.edu.sa

[1] Department of Mathematics, Ch. Charan Singh University, Meerut, 250004, Uttar Pradesh, India

[2] Department of Computer Science & Software Engineering, International Islamic University, Islamabad, Pakistan

[3] Department of Computer Science and Engineering, Xiamen Institute of Technology, Xiamen 361021, China

[4] School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

[5] Nanjing University of Information Science and Technology, Nanjing 210044, China

[6] Department of Mathematics and Computer Sciences, Kharazmi University, Tehran, Iran

[7] Center of Excellence in Information Assurance (CoEIA), King Saud University, Riyadh, Kingdom of Saudi Arabia

1 Introduction

The session initiation protocol (SIP) has gained much popularity as SIP can accomplish sessions including multimedia distribution, internet multimedia conferences and the internet telephone calls. Authentication is performed when a remote user wants to access SIP services enabling the verification of legality of both the remote user as well as the SIP server. The SIP authentication can be performed in a variety of ways for differing applications, like one time password authentication, public key cryptography and digital signatures, some other protocols for SIP authentication

are IP SEC, SSL, SSH and Kerberos. Typically the use of authentication protocol depends on the sensitivity of different applications as well as the available resources [1, 2]. The password based authenticated key agreement requires the authenticity of both the user and SIP server before initiating the session. A number of password based authentication schemes are proposed for SIP [3–26]. In earlier such schemes, the SIP server maintained a database for password of each user. And in such cases, the server has to protect the database from internal and external adversaries. Such databases are vulnerable to stolen verifier attacks as well as burdening the server resources. In 2013, Zhang et al. [27] came up with a solution to this problem, in their solution they eliminated the need of server database, and claimed their protocol to be efficient and secure, but Tu et al. [28] proved their protocol to be vulnerable to user impersonation attack. Furthermore, Tu et al. [28] proposed an enhanced protocol to improve the security, but Farash [29] questioned the security of Tu et al.'s protocol and proved it to be vulnerable to server impersonation attack. Then Farash [29] proposed an improved protocol and claimed their proposed protocol to withstand the known attacks. However in this paper, we show that Farash's protocol [29] is vulnerable to user impersonation attack, password guessing attack, session-specific temporary information leakage attack and lacks user anonymity. Furthermore, we proposed an improved protocol. The proposed protocol not only robust against all known attacks, but is also lightweight as compared to Farash's protocol [29]. Rest of the paper is organized as: Section 2 describes the basics of Elliptic curve cryptography along with related computational hard problems. Section 3 reviews Farash's scheme, and its cryptanalysis is discussed in Section 4. In Section 5, we propose our scheme. Security of the proposed scheme is analyzed in Section 6. We also performed automatic protocol verification using automated verification tool ProVerif in Section 7. A comparative performance analysis of the proposed scheme is discussed in Section 8. Finally, we give our conclusions in Section 9.

2 Preliminaries

This section describes some basics of elliptic curve cryptography(ECC) along with related hard problems to clasp the concepts used in this paper.

2.1 Elliptic curve cryptography

Miller [30] and Koblitz [31] were the first to present the use of elliptic curves in cryptography, which latterly proved to be more efficient as compared to conventional public key cryptography [32]. ECC provides same security for reduced parameters size. In elliptic curver based cryptography, the mathematical operations are carried out on an equation $E_q(i, j) : y^2 = x^3 + ix + j \mod q$, where q is a large prime number such that size of $q \geq 160\ bits$ & $i, j \in Z_p^*$. The integers i, j defines the curve, while (x, y) are the points which fulfills the statement, $4i^3 + 27j^2 \mod q \neq 0$. The point at infinity O is considered as identity element. Two common operations over $E_q(i, j)$ are point addition and scalar point multiplication, where scalar point multiplication is considered as repeated addition. Let R is a point over $E_q(i, j)$ and k is an integer then $kR = R + R + R + ...R\ (k\ times)$. The domain parameters (q, i, j, R) belongs to the finite field. Below are the two common computational problems pertaining to ECC security:

1. Elliptic Curve Discrete Logarithm Problem (ECDLP)
   ECDLP can be stated as, given two points R and S over an elliptic curve $E_q(i, j)$, it is computationally hard to find an integer k such that $R = kS$ in polynomial time.
2. Elliptic Curve Computational Diffie Hellman Problem (ECCDH)
   ECCDH can be stated as, given three points Q, aQ and bQ over an elliptic curve $E_q(i, j)$, it is computationally hard to find abQ in polynomial time.

3 Review of Farash's scheme

Here we give description of Farash's session initiation protocol which involves four phases: setup, registration, login-authentication, and password change phases. As an aid, we make Table 1 for notations meaningful in this paper.

3.1 Setup phase

The server S selects an elliptic curve E over the finite field Fq and an additive group G of order p with P as generator. S selects three one-way hash functions $h : \{0, 1\}^* \rightarrow \{0, 1\}^n, h_1 : G \times \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}^n$, and $h_2 : G \times G \times \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}^n$. S chooses a private key k, computes public key kP. Finally, S publishes the parameters $\{E(Fq), P, p, G, h, h_1, h_2\}$, and keeps k as secret key.

3.2 Registration phase

In this phase, a person who wishes to be a legal user U of the system registers itself at the server S over a secure channel.

Table 1 The notations and their meaning

Notations	Description
U	User
S	Server
A	An adversary
$username_u$	Identity of U
pw_u	Password of U
k	Private key of S
kP	Public key of S
p, q	Two prime numbers
E or $E(F_q)$, F_q	An elliptic curve, a finite field
P	The base point
G	Subgroup of $E(F_q)$ generated by P
Z_p^*	the non-zero integers modulo p
$sessionkey$	Session key between two entities
$\oplus$	Exclusive-OR operator
$\|\|$	String Concatenation operator
Enc_{key}/Dec_{key}	Encryption/Decryption

The steps involved are as follows:

1. U freely chooses his $username_u$, password pw_u, and a random number $a_u \in Z_p^*$. U computes $h(pw_u\|a_u)$ and sends the registration request $\{username_u, h(pw_u\|a_u)\}$ to S.
2. For the received registration request $\{username_u, h(pw_u\|a_u)\}$, S computes $r_u = (h(pw_u\|a_u) + h(username_u\|k))P$. S stores r_u in a smart card SC and issues it to U.
3. U inserts the random number a_u in the received SC.

3.3 Login-Authentication Phase

In this phase, U first initiates the login process and then interacts with S for mutual authentication. The steps involved are as follows:

1. U inserts his/her SC to a card reader and then inputs $username_u$ and pw_u.
2. SC chooses a random number $b \in Z_p^*$ and computes bP, $w_u = b(r_u - h(pw_u\|a_u)P)$ and $z_u = h(username_u\|bP\|w_u)$. SC sends the login request $\{username_u, bP, z_u\}$ to S.
3. For the received login request $\{username_u, bP, z_u\}$, S computes $w_u^* = h(username_u\|k)$ bP and $z_u^* = h(username_u\|bP\|w_u^*)$. Then S compares z_u^* and z_u. If $z_u^* = z_u$, U is authenticated and S selects two random numbers $c, d \in Z_p^*$,computes dP, $v = d(bP)$, $sessionkey = h_1(v\|c\|username_u)$ and $Auth_s = h_2(v\|w_u^*\|c\|sessionkey)$. S sends challenge request $\{realm, dP, c, Auth_s\}$ to U.
4. For the received challenge request $\{realm, dP, c, Auth_s\}$, U computes $v = b(dP)$, $sessionkey = h_1(v\|c\|username_u)$, $Auth_s^* = h_2(v\|w_u\|c\|sessionkey)$. U compares $Auth_s^*$ and $Auth_s$. If $Auth_s^* = Auth_s$, S is authenticated. Then U computes $Auth_u = h_2(v\|z_u||c+1\|sessionkey)$ to send the challenge response $\{realm, Auth_u\}$to S.
5. For the received challenge response $\{realm, Auth_u\}$, S computes $Auth_u^* = h_2(v\|zu^*\|c+1\|sessionkey)$. S compares $Auth_u^*$ and $Auth_u$. If $Auth_u^* = Auth_u$, U is re-authenticated with the assurance of no replay.

3.4 Password changing phase

In this phase, U can change his/her password with the help of the server S. For this, U first establishes a $sessionkey$ with S by undergoing the login-authentication process. Then U executes the following steps to change his/her password:

1. U chooses a new password pw_{unew} and two new random numbers $a_{unew}, e \in Z_p^*$. Then U computes $m_u = Enc_{sessionkey}(username_u\|e\|h(pw_{unew}\|a_{unew})\| h(username_u\|e\|h(pw_{unew}\|a_{unew})))$. Then U sends the password change request $\{m_u, e\}$ to S.
2. For the received password change request $\{m_u, e\}$, S decrypts m_u to obtain the embedded values $username_u, e, h(pw_{unew}\|a_{unew})$, $h(username_u\|e\|h(pw_{unew}\|a_{unew}))$. S checks the validity of $h(username_u\|e\|h(pw_{unew}\|a_{unew}))$. If it passes the validity test, S computes $r_{unew} = (h(pw_{unew}\|a_{unew}) + h(username_u\|k))P$ and $m_s = Enc_{sessionkey}(r_{unew}\|h(username_u\|e+1\|r_{unew}))$. S sends $\{m_s\}$ to U.
3. For the received $\{m_s\}$, U decrypts m_s as $Dec_{sessionkey}(m_s)$ to obtain the embedded values r_{unew} and $h(username_u\|e+1\|r_{unew})$. U checks the validity of $h(username_u\|e+1\|r_{unew})$.If it passes the validity test, U replaces r_{unew} and a_{unew} with r_u and a_u respectively.

4 Cryptanalysis of Farash's scheme

In this section, we explain the vulnerabilities of Farash's scheme. Our cryptanalysis is based on an adversary's capability to intercept and transmit messages over the open channel, and extract values stored in a smart card [33, 34].

4.1 User impersonation attack

An adversary A can successfully impersonate a valid user to login the server in the following manner:

1. A obtains the $username_u$ of U by intercepting the login request $\{username_u, bP, z_u\}$ of U. A uses username$_u$ to achieve the secret parameter $h(username_u||k)P$ of U. For this A selects a simple value t_A, computes its hash output $h(t_A)$ and sends $username_u, h(t_A)$ to S. In response, A obtains a smart card containing $r_{uA} = (h(t_A) + h(username_u||k))P$. A extracts r_{uA} from his/her smart card and computes $r_{uA} - h(t_A)P = h(username_u||k)P$.
2. Then A proceeds to impersonate U. A chooses a random number $b_A \in Z_p^*$, computes b_AP, $w_{uA} = b_Ah(username_u||k)P$ and $z_{uA} = h(username_u||b_AP||w_{uA})$. SC sends the login request $\{username_u, b_AP, z_{uA}\}$ to S.
3. For the received login request, S computes $w^*_{uA} = h(username_u||k)b_AP$ and $z^*_{uA} = h(username_u||b_AP||w^*_{uA})$. S compares z^*_{uA} and z_{uA}. It is clear that $w^*_{uA} = w_{uA}$, therefore $z^*_{uA} = z_{uA}$ and hence S believes to be connected with the valid user U. Then S selects two random numbers $c, d \in Z_p^*$, and computes dP, $v_A = d(b_AP)$, $sessionkey_A = h_1(v_A||c||username_u)$ and $Auth_A = h_2(v_A||w^*_{uA}||c||sessionkey_A)$. S sends challenge request $\{realm, dP, c, Auths_A\}$.
4. A intercepts and blocks the challenge request $realm, dP, c, Auth_{sA}$. A computes $v_A = b_A(dP)$ and $sessionkey_A = h_1(v_A\|c\|username_u)$. A also computes $Auth_A = h2(v_A\|z_{uA}\|c + 1\|sessionkey_A)$ and sends the challenge response $\{realm, Auth_A\}$ to S.
5. For the received challenge response $\{realm, Auth_A\}$, S computes $Auth^*_A = h2(v_A\|z^*_{uA}\|c + 1\|sessionkey_A)$. S compares $Auth^*_A$ and $Auth_A$. It is clear that $z^*_{uA} = z_{uA}$. Now both A and S compute the same value of v_A, therefore $Auth^*_A = Auth_A$. Therefore, S is assured that the connected entity is the valid user U and believes $sessionkey_A$ is shared with U.

In this way, A impersonates a legal user U to login S and also establishes $sessionkey_A$ with the server S.

4.2 Lacks user anonymity

User U sends the login request $\{username_u, bP, z_u\}$ which contains U's identity $username_u$ in plaintext. Thus, an adversary A can easily pick $username_u$ from the network and can misuse it. Presence of plaintext identity of a user in login request reveals user related information like login-frequency. Thus, A can have an idea of the purpose behind login. As a result, A can harm a legal user. But, Farash's scheme overlooks these possibilities as it does not provide user anonymity.

4.3 Password guessing attack

If A finds the lost smart card of U then he can extract the values $\{r_u, a_u\}$ from it. Further, he can proceed to guess U's password. A guesses pw_p as U's possible password and computes $h(pw_p||a_u)$. A obtains $username_u$ of U by intercepting the login request $\{username_u, bP, zu\}$ from the network. A chooses a random number $b_A \in Z_p^*$, computes b_AP, $w_{uA} = b_A(r_u - h(pw_p||a_u)P)$ and $z_{uA} = h(username_u||b_AP||w_{uA})$. A sends the login request $username_u, b_AP, z_{uA}$ to S. If A receives challenge response from S, then the guessed password pw_p is correct otherwise A tries with some other guess.

4.4 Session-specific temporary information attack

According to Canetti and Krawczyk [35], this attack targets to compute the established session key of a particular session in case the random numbers of this session are leaked. In Farash's scheme, suppose that the random number d is leaked. Then, an adversary A can take bP from an intercepted login request $\{username_u, bP, z_u\}$ of U. Further, A can compute $v = d(bP)$ and hence he can compute the session key $sessionkey = h_1(v||c||username_u)$. In this way, $sessionkey$ is vulnerable under the leakage of session-specific temporary information.

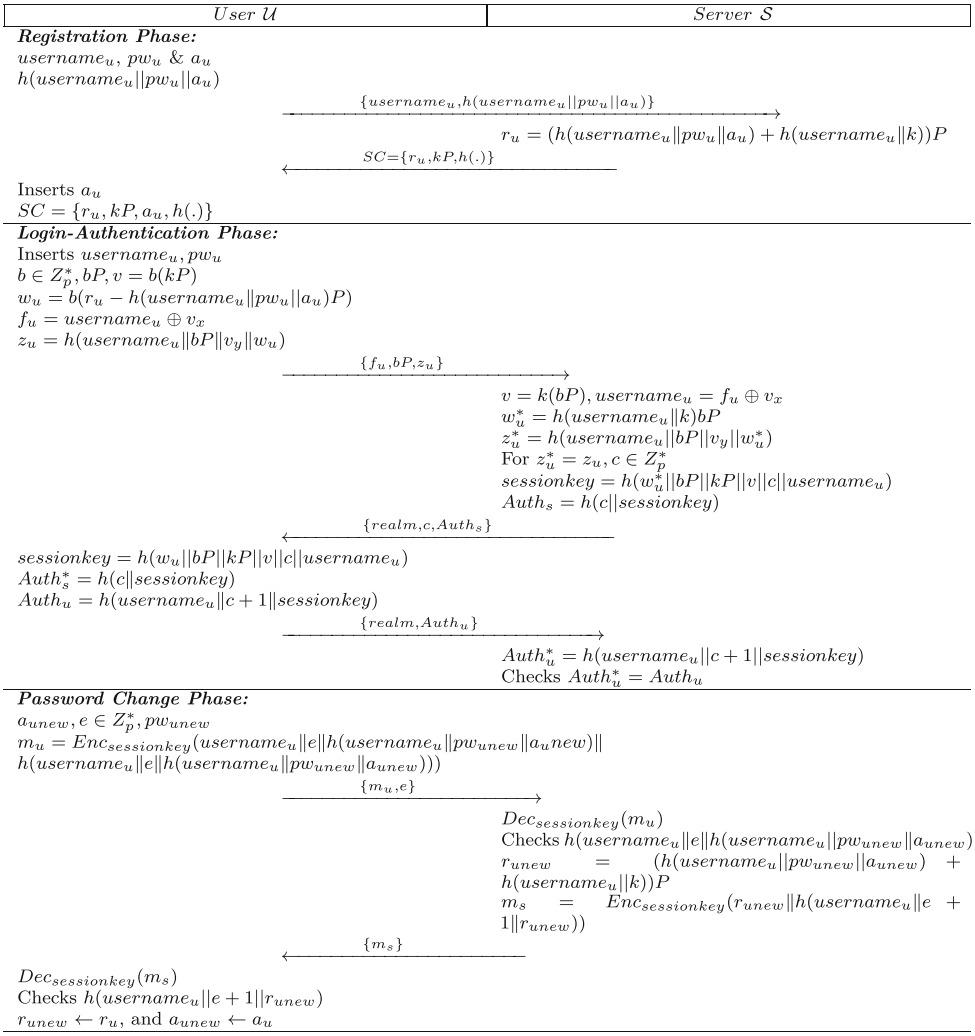
5 Proposed scheme

The proposed scheme consists of four phases: setup, registration, login-authentication, and password change phases. As an aid, we make Fig. 1 of the proposed scheme. The reason for the aforementioned vulnerabilities of Farash's scheme is transmission of user's plaintext identity $username_u$ during login request and the design of user-specific value as $h(pw_u||a_u)$. As a solution, we have hidden the identity $username_u$ of U in such a way that only the valid server can obtain the real identity of the user. Besides, we have modified the design of user-specific value $h(pw_u||a_u)$ to $h(username_u\|pw_u\|a_u)$.

5.1 Setup phase

The server S selects an elliptic curve E over the finite field Fq and an additive group G of order p with P as generator. S selects a one-way hash functions $h(.)$. S chooses a private key k, computes public key kP. Finally, S publishes the parameters $\{E(Fq), P, p, G, h(.)\}$, and keeps k as secret key.

Fig. 1 The Proposed Scheme



5.2 Registration phase

In this phase, a person who wishes to be a legal user U of the system registers itself at the server S over a secure channel. The steps involved are as follows:

1. U freely chooses his $username_u$, password pw_u, and a random number $a_u \in Z_p^*$. U computes $h(username_u||pw_u||a_u)$ and sends the registration request $\{username_u, h(username_u\|pw_u\|a_u)\}$ to S.
2. For the received registration request $\{username_u, h(username_u\|pw_u\|a_u)\}$, S computes $r_u = (h(username_u\|pw_u\|a_u) + h(username_u\|k))P$. S stores r_u in a smart card and issues $SC = \{r_u, kP, h(.)\}$ to U.
3. U inserts the random number a_u in received SC, thus SC contains $\{r_u, kP, a_u, h(.)\}$.

5.3 Login-authentication phase

In this phase, U first initiates the login process and then interacts with S for mutual authentication. The steps involved are as follows:

1. U inserts his/her SC to a card reader and then inputs $username_u$ and pw_u.
2. SC chooses a random number $b \in Z_p^*$, computes bP, $v = b(kP)$ and $w_u = b(r_u - h(username_u||pw_u||a_u)P)$. SC also computes $f_u = username_u \oplus v_x$ and $z_u = h(username_u\|bP\|v_y\|w_u)$ where v_x and v_y are x^{th} and y^{th} components of v respectively. SC sends the login request $\{f_u, bP, z_u\}$ to S.
3. For the received login request $\{f_u, bP, z_u\}$, S computes $v = k(bP)$ to retrieve $username_u = f_u \oplus v_x$. Then S computes $w_u^* = h(username_u\|k)bP$

and $z_u^* = h(username_u \| bP \| v_y \| w_u^*)$. S compares z_u^* and z_u. If $z_u^* = z_u$, U is authenticated and S selects a random number $c \in Z_p^*$, S computes $sessionkey = h(w_u^* \| bP \| kP \| v \| c \| username_u)$ and $Auth_s = h(c||sessionkey)$. S sends challenge request $\{realm, c, Auth_s\}$to U.

4. For the received challenge request $\{realm, c, Auth_s\}$, U computes $sessionkey = h(w_u \| bP \| kP \| v \| c \| username_u)$, $Auth_s^* = h(c \| sessionkey)$. U compares $Auth_s^*$ and $Auth_s$. If $Auth_s^* = Auth_s$, S is authenticated and U computes $Auth_u = h(username_u \| c + 1 \| sessionkey)$ to send the challenge response $\{realm, Auth_u\}$ to S.
5. For the received challenge response message $\{realm, Auth_u\}$, S computes $Auth_u^* = h(username_u \| c + 1 \| sessionkey)$. S compares $Auth_u^*$ and $Auth_u$. If $Auth_u^* = Auth_u$, U is re-authenticated with the assurance of no replay.

5.4 Password changing phase

In this phase, U can change his/her password with the help of the server S. For this, U first establishes a $sessionkey$ with S by undergoing the login-authentication process. Then U executes the following steps to change his/her password:

1. U chooses a new password pw_{unew} and two new random numbers $a_{unew}, e \in Z_p^*$. Then U computes $m_u = Enc_{sessionkey}(username_u \| e \| h(username_u \| pw_{unew} \| a_{unew}) \| h(username_u \| e \| h(username_u \| pw_{unew} \| a_{unew})))$. Further U sends the password change request $\{m_u, e\}$ to S.
2. For the received request $\{m_u, e\}$, S decrypts m_u to obtain the embedded values. Then S checks the validity of $h(username_u \| e \| h(username_u \| pw_{unew} \| a_{unew}))$. If it passes the validity test, S computes $r_{unew} = (h(username_u \| pw_{unew} \| a_{unew}) + h(username_u \| k))P$ and $m_s = Enc_{sessionkey}(r_{unew} \| h(username_u \| e + 1 \| r_{unew}))$. S sends $\{m_s\}$ to U.
3. For the received m_s, U decrypts m_s as $Dec_{sessionkey}(m_s)$ to obtain the embedded values r_{unew} and $h(username_u \| e + 1 \| r_{unew})$. U checks the validity of $h(username_u \| e + 1 \| r_u new)$. If it passes the validity test, U replaces r_{unew} and a_{unew} with r_u and a_u respectively.

6 Security analysis

We have performed formal as well as informal security analysis of our proposed scheme in following subsections.

6.1 Provable security model and proof

In this subsection, we prove that our scheme is secure with a formal security model based on [36, 37].

6.1.1 Security model

To make the discussion simple, we suppose that only two participants are in our protocol $\mathcal{P}$: a user U and a server S. In the executing process, both U and S have many instances. Each instance is with a number k and it is an oracle. One instance is an execution of $\mathcal{P}$. We define U^i or S^j as the instance for U or S with its own number i and j, respectively, or I^k with differences eliminated. In this proof, three possible states can be the result of an oracle: accept, reject or $\perp$. When the oracle receives a correct message, it reaches the accept state. Otherwise it reaches the reject state. If no decision has been reached or at last no result has been returned, the state $\perp$ appears.

Before the instances start, U owns the identity $username_u$, a password pw_u, the generator point P and a smart card containing r_u, kP and a_u. S has the private key k and the public key kP. The number of passwords is finite. And the passwords are in a special dictionary $\mathcal{D}$ with total number $|\mathcal{D}|$. $\mathcal{P}$ can be executed many times by each instance. In this proof we consider the server S as secure.

According to the adversary's model, the attacker A completely controls the public channel, and he aims to break privacy of the communication or the session keys. A can make some queries on oracles and get answers. We list the queries below:

- $h(str)$: This is the hash oracle. We suppose that the result is r. A record (str, r) is formed after that query. How to deal with the record is in the proof process.
- $Send(U^i/S^j, m/INIT)$: This query models A's active actions in communication, and will output the message that the instance would generate once receiving message m. If the second parameter is $INIT$, the result is the first message (f_u, bP, z_u) in the step 2 of Login-Authentication phase. The query is finished like the steps in the scheme.
- $Execute(U^i, S^j)$: It models passive attacks in the channel. A can obtain messages between U^i and S^j in normal executions. The query outputs the transcripts in execution.
- $Reveal(I^k)$: This query denotes known-key attacks. After I^k computes a session key, A can obtain it by this query.
- $Corrupt(smartcard)$: A gets all data in U's smart card after this query. It is for the guessing attack.
- $Test(I^k)$: This query is corresponding for gaining the session key. After multiple queries, A should select a

session to challenge. If no session key is generated for the instance I^k, return $\perp$. Otherwise, a coin ω will be flipped. If $\omega = 1$, the real session key is returned; if $\omega = 0$, a random string as long as the session key is returned. This query can be only asked on the $fresh$ instance once. The notion $fresh$ will be introduced below.

Here we give some definitions to show the security of the scheme:

- $Partnering$: Each instance U^i or S^j has a partner identity pid_U^i or pid_S^j, a session key sk_U^i or sk_S^j and a session identity sid_U^i or sid_S^j once it is accepted and forms a session key. U^i and S^j are partners if and only if $pid_U^i = S^j$, $pid_S^j = U^i$, $sid_U^i = sid_S^j$ and $sk_U^i = sk_S^j$.
- $fresh$: The instance I^k is thought to be $fresh$ while $Reveal(I^k)$ or $Reveal(pid_I^k)$ does not happen:
- $AKE - security$: The advantage of the adversary A breaking $\mathcal{P}$ is the probability which A correctly guesses the coin ω produced in $Test(I^k)$ with I^k accepted and $fresh$. If A outputs $\omega^{'}$, the advantage is defined as

  $$Adv_{\mathcal{P}}^{AKE}(A) = |2Pr[\omega = \omega^{'}] - 1|$$

  Our scheme is $AKE{-}secure$ if $Adv_{\mathcal{P}}^{AKE}(A)$ is negligibly greater than $\frac{O(q_s)}{|\mathcal{D}|}$, depending on security parameter l_s. Here q_s is the query times of $Send$.

Moreover, we list some computation assumptions.

- Elliptic curve computational Diffie-Hellman (ECCDH) problem assumption: Given $\alpha P, \beta P \in G$ and α, β are positive integers, A can calculate $\alpha\beta P$ in polynomial time t. We define the probability A can solve the problem in t as $Adv_A^{ECCDH}(t)$ and now $Adv_A^{ECCDH}(t) \leq \epsilon$ where ϵ is a negligibly positive number.

6.1.2 Security proof

Theorem 1 *The user employs password from dictionary $\mathcal{D}$ with size $|\mathcal{D}|$. l_s denotes the security length, which is for hash values. $\mathcal{P}$ is our scheme. For an attacker A running with time t at most makes q_s $Send$ queries, q_e $Execute$ queries and q_h hash oracle queries, we have*

$$Adv_{\mathcal{P}}^{AKE}(A) \leq \frac{(q_s+q_e)^2}{p-1} + \frac{q_h^2}{2^{l_s}} + \frac{3q_s+2q_h}{2^{l_s-1}} + \frac{2q_s}{|\mathcal{D}|} + 4q_h Adv_A^{ECCDH}(t + (5q_e + 3q_s)t_{epm})$$

where t_{epm} is the computation time of one scalar multiplication in G.

Proof We define a series of games from Game G_0 to G_4. $Succ_i$ is event that A guesses ω for G_i successfully in $Test$. According to premises of our model, the attacker need not guess or compute the user's identity due to only one user. The games are listed as follows:

- Game G_0: This is the real scheme with the random oracle model. We see that $Adv_{\mathcal{P}}^{AKE}(A) = 2Pr[Succ_0] - 1$. We choose a random bit $\omega^{'}$ if the game aborts, stops without getting answer from A or A has not completed the games because he uses more queries or more time than predetermined upper bounds.
- Game G_1: We simulate all oracles for queries. Also, three lists are used to store the record (str, r) formed after query mentioned in the security model. L_h stores the answers to hash oracles. While the h oracle is queried by A, the record is in L_A. And $L_{\mathcal{P}}$ is for transcripts between U and S. We can not distinguish G_0 and G_1 via the simulation, so $Pr[Succ_1] = Pr[Succ_0]$.
- Game G_2: Some collisions should be avoided on the transcripts. b and c may be the same points in different transcripts. Also, hash values may collide. Note that $b, c \in [1, p-1]$ and the lengths of hash values are l_s. G_2 and G_1 are indistinguishable unless the above collisions appear. According to birthday paradox, we can see that

  $$|Pr[Succ_2] - Pr[Succ_1]| \leq \frac{(q_s+q_e)^2}{2(p-1)} + \frac{q_h^2}{2^{l_s+1}}$$

- Game G_3: In this game, we abort the game if A has luckily calculated correct messages without corresponding hash oracles. We divide the game into three cases according to three messages.

  1. To forge $Send(S^j, (f_u, bP, z_u))$ query, A must make $(username_u||bP||v_y||w_u, z_u)$ hash query. Or we can say that $(username_u||bP||v_y||w_u, z_u) \in L_A$ should be true. If we have not found it as a role of server, the probability is up to $\frac{q_s}{2^{l_s}}$. Note that S does not know pw_u, so the record $(username_u||pw_u||a_u, *)$ can not be checked. The probability is $\frac{q_h}{2^{l_s}}$.
  2. To forge $Send(U^i, (realm, c, Auth_s))$, A must make $(w_u||bP||kP|| * ||c||username_u, *)$ and $(c||*, Auth_s)$. The probabilities are upper bounded by $\frac{q_h}{2^{l_s}}$ and $\frac{q_s}{2^{l_s}}$ respectively for the matter that the two records do not exist in L_A .
  3. To forge $Send(S^j, (realm, Auth_u))$, A must make $(username_u||c+1||*, Auth_u)$ and it is bounded by $\frac{q_s}{2^{l_s}}$ for the matter that the record does not exist in L_A.

  So the two games G_3 and G_2 are indistinguishable unless the messages are forged without hash queries. So we have

  $$|Pr[Succ_3] - Pr[Succ_2]| \leq \frac{3q_s+2q_h}{2^{l_s}}$$

– Game G_4: In this game, ECCDH problem is brought in. A can use random oracles normally. If A can successfully gain the session key $sessionkey$ and win the game, we think that we use A to solve the ECCDH problem. If A can compute the session key, he must ask a $(w_u||bP||kP||v||c||username_u, sessionkey)$ query. If the above record correctly exists in the list L_A, A breaks the ECCDH problem.

  We call this event $Guessing$. It is clear to see that

  $$|Pr[Succ_4] - Pr[Succ_3]| = Pr[Guessing]$$

  We divide $Guessing$ into two cases: online guessing attack and off-line guessing attack.

  – Case 1: Suppose A uses $Corrupt(smartcard)$ to get the data in U's smart card before online guessing. Since there are q_s $Send$ queries, the probability for this case is $\frac{q_s}{|\mathcal{D}|}$.
  – Case 2: It is for off-line guessing attack. First A uses $Corrupt(smartcard)$, then the execution process are done by A. The record $(w_u||bP||kP||v||c||username_u, sessionkey)$ is in L_A with the probability $\frac{1}{q_h}$. There are two ways for the aim. One is A asks $Execute$ query, and the other is A asks $Send$ queries in order like an $Execute$ query. The probabilities for them are $q_h Adv_A^{ECCDH}(t + 5q_e t_{epm})$ and $q_h Adv_A^{ECCDH}(t + 3q_s t_{epm})$ respectively.

  From the above analysis, we can see G_4 and G_3 are indistinguishable unless $Guessing$ happens. It seems that

$$\begin{aligned} |Pr[Succ_4] - Pr[Succ_3]| &\leq \frac{q_s}{|\mathcal{D}|} + q_h Adv_A^{ECCDH} \\ &\quad \times (t + 5q_e t_{epm}) \\ &\quad + q_h Adv_A^{ECCDH}(t + 3q_s t_{epm}) \\ &\leq \frac{q_s}{|\mathcal{D}|} + 2q_h Adv_A^{ECCDH} \\ &\quad \times (t + (5q_e + 3q_s)t_{epm}) \end{aligned}$$

Until now, A has no advantage in guessing ω and $Pr[Succ_4] = \frac{1}{2}$. So the expression in Theorem 1 can be calculated. □

6.2 Further security discussion

6.2.1 Provides user anonymity and user un-traceability

In our scheme, user's smart card stores $r_u, kP, a_u, h(.)$ which does not contain the plaintext identity of the user. The value $r_u = (h(username_u \| pw_u \| a_u) + h(username_u \| k))P$. An adversary A cannot obtain U's identity $username_u$ from r_u due to both ECDLP and the one-way property of hash function. The login request f_u, bP, z_u also does not provide $username_u$ of U directly. A cannot recover the random number b of U using bP due to ECDLP, therefore, A cannot compute v_x to gain $username_u$ from f_u. Further, A cannot recover $username_u$ from $z_u = h(username_u \| bP \| v_y \| w_u)$ due to the one-way property of hash function. Any two login requests of U are totally different from each other as every time fresh values are computed using freshly chosen random numbers. As a result, A cannot trace a user by watching its login requests on the network. Hence, our scheme provides user anonymity and user un-traceability.

6.2.2 Resists user impersonation attack

For impersonating a legal user U, the knowledge of U's identity $username_u$ and the related secret value $h(username_u \| k)$ is essential. But the proposed scheme provides security against identity revelation as described in Section 6.2.1. A cannot obtain the secret parameter $h(username_u \| k)P$ of U by proceeding as in Farash's scheme because there is no way for A to gain $username_u$ of U. A can extract a_u from stolen smart card of u but it is not possible to guess two values $username_u$ and pw_u simultaneously in real time polynomial. Thus, A cannot retrieve $h(username_u \| k)P$ from r_u. A can select a new random number $b_A \in Z_p^*$ and can compute $v_A = b(kP)$. However, computation of $f_u = username_u \oplus vx$ and $z_{uA} = h(username_u \| b_A P \| v_y \| b_A h(username_u \| k)P)$ is not feasible without the correct $username_u$ and $h(username_u \| k)$. Since A is not capable of computing a workable login request to cheat S, user impersonation is not possible in the proposed scheme.

6.2.3 Resists password guessing attack

A can steal U's smart card and can extract the values r_u, kP and a_u from it. A can guess pw_p as the possible password of U and can use a_u, he cannot verify the correctness of his guess using r_u in the absence of correct $username_u$ and secret value $h(username_u \| k)$. The identity $username_u$ of U is always embedded in other values transmitted over open network such that no one except the valid server can obtain it. A is not capable of obtaining neither $username_u$ nor $h(username_u \| k)$ of U as discussed in Sections 6.2.1 and 6.2.2. In the lack of correct $username_u$, A has no way to verify the accuracy of the guessed pw_p by sending a trial login request to S and waiting for its response. As a result, password guessing is not feasible in our scheme as in Farash's scheme. Thus, the proposed scheme is safe against password guessing attack.

6.2.4 Resists replay attack

An adversary A can replay the login request f_u, bP, z_u of U to S. After receiving the challenge response from S, A needs to reply with correct $Auth_u$ which requires the knowledge of $username_u, h(username_u \| k)$ and random number b. It is not feasible to recover $username_u$ and $h(username_u \| k)$ as discussed in Sections 6.2.1 and 6.2.2, so the value $h(username_u || k)bP$ cannot be computed by A. Random number b cannot be gained from bP due to ECDLP. Without having b, computation of $b(kP)$ is not possible. Thus, A cannot compute the correct $sessionkey$ corresponding to the session the login request of which is replayed. Due to lack of $username_u, h(username_u \| k)bP$ and $sessionkey$, A cannot compute the correct response, therefore, he cannot clear the final authentication test in last step of the login-authentication phase. Thus, the replay of a login request is not useful for A and replay attack is not applicable in the proposed scheme.

6.2.5 Provides mutual authentication

When U sends the login request f_u, bP, z_u, S authenticates U by itself computing $z_u^* = h(username_u \| bP \| v_y \| w_u^*)$ and comparing it with the received z_u. For this, S itself computes $v = k(bP)$ using its private key k, retrieves $username_u = f_u \oplus v_x$ and again uses k to compute $w_u^* = h(username_u \| k)bP$. The equality of z_u^* and z_u validates U. When S sends the challenge request $realm, c, Auth_s, U$ authenticates S by computing $Auth_s^* = h(c||sessionkey)$ and comparing it with the received $Auth_s$. For this, U computes $sessionkey = h(wu * \|bP\|kP\|v\|c\|usernameu)$ using his/her $username_u$ and already computed $v = b(kP)$. The equality of $Auth_s^*$ and $Auth_s$ validates S. Finally, S receives $realm, Auth_u$ and computes $Auth_u^* = h(username_u \| c + 1 \| sessionkey)$ using already computed $sessionkey$, w_u^* and retrieved $username_u$. Then S compares $Auth_u^*$ with the received $Auth_u$, the equality of these two values ensures that the legitimate user U is connected and the login request was not replayed by an adversary. In this way, the proposed scheme provides mutual authentication.

6.2.6 Resists man-in-the-middle attack

In this attack, an adversary A sits in the middle of the conversation of a user and the server, intercepts and blocks the messages transmitted between these two legal entities, itself connects with them and makes them believe that they are connected with each other. When U transmits the login request f_u, bP, z_u to S, A can intercept and block this request. But A cannot compute $v = k(bP)$ similar to $b(kP)$ computed by U as it requires the knowledge of private key k of S. Further, A requires $username_u$ and $h(username_u||k)$ to communicate with U acting as S and with S acting as U. Consequently, A cannot apply man-in-the-middle on the proposed scheme.

6.2.7 Resists session-specific temporary information attack

Assume that the random numbers b and c are leaked in our scheme. Then, A can compute $v = b(kP)$ using the public key kP of the server, so A holds v_x and v_y. A can use f_u from an intercepted login request f_u, bP, z_u to retrieve $username_u$ as $username_u = f_u \oplus v_x$. Thus, A obtains the values bP, c & $username_u$ to compute $sessionkey$ and kP is the public key of the server but the value $h(username_u||k)bP$ is missing. Although A holds b & $username_u$ and P is public parameter but A cannot compute $h(username_u||k)bP$ due to involvement of the private key k of S. As a result, A cannot compute $sessionkey = h(h(username_u||k)bP||bP||kP||v||c||username_u)$ in spite of the compromise of random numbers b and c. Therefore, our scheme is free from session-specific temporary information attack.

7 Automated security verification using ProVerif

ProVerif is a toolkit used for verifying security properties for cryptographic protocols using a specification language based on an extension of pure Pi-calculus. We use ProVerif to prove that proposed scheme satisfies the mutual authentication and session key secrecy [38–41]. ProVerif supports a number of cryptographic primitives, including encryption and decryption (symmetric and asymmetric), digital signatures, and hash function, Diffie-Hellman key agreements, and so on. Initially we defined two channels, a secure channel SCh1 is used for the secure communication between user and the Server, and a public channel PCh2 is used for the public /insecure communication between user and the Server.

```
(* ————- Channels ————————*)
free SCh1:channel [private]. (*secure channel between User and Server *)
free PCh2:channel. (*public channel between User and Server *)
```

In this security analysis, session key is modeled as follows:

```
(* ————-Session Key————————*)
free sessionkey:bitstring [private].
```

All public parameters are defined as follows:

```
(*————— Variables & Constants ————————*)
free usernameu:bitstring.
free sessionkey:bitstring [private].
free pwu:bitstring [private].
```

```
const k:bitstring [private].
const p:bitstring.
const q:bitstring.
const P:bitstring.
```

One-way Hash, Multiplication, Concatenation, Exclusive OR, getx, gety, addition, Elliptic Curve Point Multiplication and Subtraction functions are modeled as constructors.

```
(*————————— Constructors —————————*)
fun oneWH(bitstring):bitstring.
fun getx(bitstring):bitstring.
fun gety(bitstring):bitstring.
fun concat(bitstring,bitstring):bitstring.
fun add(bitstring,bitstring):bitstring.
fun ExcOR(bitstring,bitstring):bitstring.
fun multi(bitstring,bitstring):bitstring.
fun ECMP(bitstring,bitstring):bitstring.
fun subtract(bitstring,bitstring):bitstring.
```

We also defined the following equation to benefit exclusive or property $(a \oplus b) \oplus b = a$.

```
(*————————— Destructors & Equations —————————*)
equation forall a:bitstring,b:bitstring; ExcOR(ExcOR(a,b),b)=a.
```

Following four events are defined to model the initiation and termination of both user and server processes.

```
(*————————— Events —————————*)
event User_begin(bitstring).
event User_end(bitstring).
event Server_begin(bitstring).
event Server_end(bitstring).
```

Following three queries are applied, attacker(sessionkey) can verify the secrecy of the session key, while other two queries verifies the initiation and termination of both user and server events.

```
(*————————queries————————*)
query attacker(sessionkey).
query id:bitstring; inj-event(User_end(id)) ==> inj-event(User_begin(id)) .
query id:bitstring; inj-event(Server_end(id)) ==> inj-event(Server_begin(id)) .;
```

We have modeled following two processes, one for user (procUsr) and one for server (procSrv).

```
(*————————-processes————————*)
let procUsr=
in(PCh2,xkP:bitstring);
(* Registration *)
new au:bitstring;
let HunPa = oneWH(concat(usernameu,(pwu,au))) in
out(SCh1,(usernameu,HunPa));
in(SCh1,xSC:bitstring);
let SC = concat(xSC,au)in
(* Login *)
event User_begin (usernameu);
new b:bitstring;
let (xru:bitstring) = SC in
let bP = multi(b,P) in
let v = ECMP(b,xkP) in
let wu=multi(b,subtract(xru,ECMP(oneWH(concat(user nameu,(pwu,au))),P))) in
let fu = ExcOR(usernameu,getx(v)) in
let zu = oneWH(concat(usernameu,(bP,gety(v),wu))) in
out(PCh2,(fu,bP,zu));
in(PCh2,(xc:bitstring,xAuths:bitstring));
let sessionkey = oneWH(concat(wu,(bP,xkP,v,xc))) in
let Auths=oneWH(concat(xc,sessionkey)) in
if(Auths = xAuths) then
event User_end(usernameu) else 0.
let procSrv=
let kP = ECMP(k,P) in
out(PCh2,kP);
(* Registration *)
in(SCh1, (xusernameu:bitstring,xHunPa:bitstring) );
let ru = multi(add(xHunPa,oneWH(concat(xusernameu,k))),P )in
let SC = concat(ru,kP)in
out(SCh1,SC);
(* Login *)
in(PCh2,(xfu:bitstring,xbP:bitstring,xzu:bitstring));
```

Table 2 Comparison of efficiency: computational cost/complexity

Scheme→ Phases↓	Zhang et al.	Tu et al.	Farash	Proposed
Registration (U/SC)	$1t_{has}$	$1t_{has}$	$1t_{has}$	$1t_{has}$
Registration (S)	$1t_{epm} + 1t_{min}$ $+1t_{has}$	$1t_{epm} + 1t_{has}$	$1t_{epm} + 1t_{has}$	$1t_{epm} + 1t_{has}$
Password update phase (U/SC)	$2t_{sym} + 3t_{has}$	$2t_{sym} + 3t_{has}$	$2t_{sym} + 3t_{has}$	$2t_{sym} + 3t_{has}$
Password update phase (S)	$1t_{epm} + 2t_{sym}$ $+1t_{min} + 3t_{has}$	$1t_{epm} + 2t_{sym}$ $+3t_{has}$	$1t_{epm} + 2t_{sym}$ $+3t_{has}$	$1t_{epm} + 2t_{sym}$ $+3t_{has}$
Login-authentication phase (U/SC)	$5t_{epm} + 1t_{epa}$ $+6t_{has}$	$3t_{epm} + 1t_{epa}$ $+5t_{has}$	$3t_{epm} + 1t_{epa}$ $+5t_{has}$	$3t_{epm} + 1t_{epa}$ $+5t_{has}$
Login-authentication phase (S)	$4t_{epm} + 2t_{epa}$ $+4t_{has}$	$3t_{epm} + 4t_{has}$	$3t_{epm} + 4t_{has}$	$2t_{epm} + 5t_{has}$
Total computational complexity in login-authentication phase	$9t_{epm} + 3t_{epa}$ $+10t_{has}$	$6t_{epm} + 1t_{epa}$ $+10t_{has}$	$6t_{epm} + 1t_{epa}$ $+10t_{has}$	$5t_{epm} + 1t_{epa}$ $+10t_{has}$

```
event Server_begin(k);
let v = ECMP(k,xbP) in
let usernameu' = ExcOR(xfu,getx(v)) in
let wu' = ECMP(oneWH(concat(usernameu',k)),xbP) in
let zu' = oneWH(concat(usernameu',(xbP,gety(v),wu'))) in
if(xzu = zu') then
new c:bitstring;
let sessionkey = oneWH(concat(wu',(xbP,kP,v,c))) in
let Auths=oneWH(concat(c,sessionkey)) in
out(PCh2,(c,Auths));
event Server_end(k)
else 0.
process ((!procUsr)|(!procSrv))
```

We execute the modeled processes in ProVerif 1.88 (the newest version). Following are the results:

1. inj-event(Server_end(id)) ==> inj-event(Server_begin(id)) is true.
2. inj-event(User_end(id_1964)) ==> inj-event(User_begin(id_1964)) is true.
3. not attacker(sessionkey[]) is true.

The results (1) and (2) verifies that both server and user processes started and terminated successfully, while (3) verifies that sessionkey is not revealed to adversary and secrecy is maintained.

8 Comparative analysis

We compare the performance of our scheme with Zhang et al.'s [27], Tu et al.'s [28], and Farash's [29] schemes. Tables 2 and 3 display phase-wise computational cost/complexity and security features of these schemes respectively. We do not consider very lightweight operations such as XOR and string concatenation operations which contribute negligible computational cost. In Table 2, t_{epm} denotes the time complexity of elliptic curve scalar point multiplication, t_{sym} denotes the time complexity of symmetric key encryption/decryption, t_{min} denotes the time complexity of a modular inversion, t_{epa} denotes the time complexity of an elliptic curve point addition, t_{has} denotes the time complexity of a one-way hash function.

During registration phase, user computes one hash function in each of the four schemes. In the same phase, computational complexity on S is same in Tu et al.'s, Farash's and our scheme whereas in Zhang et al.'s scheme S also requires to compute a modular inversion. In password change phase, the computational load of at both, user and the server side is same in all the four schemes except one modular inversion extra at S in Zhang et al.'s scheme. During login-authentication phase, highest computational load is on Zhang et al.'s scheme. Other two schemes [28, 29] and our scheme have almost same computational load, the only change is the addition of one hash operation and reduction of one elliptic curve scalar point multiplication on the server side. This is also exhibited by the total computational complexity during login-authentication. Since hash operation is quite lightweight as compared to elliptic curve scalar point multiplication, our scheme has slightly less computational cost than in schemes [28, 29].

There is remarkable difference in the security features offered by these schemes. Zhang et al.'s scheme with highest computational load suffers from user impersonation, password guessing attacks and does not provide user anonymity. Farash proposed improvement of Tu et al.'s scheme without adding any computational operation and succeeded in removing man-in-the-middle attack. However, both the schemes [28, 29] suffer from user impersonation, password guessing, session-specific temporary information attacks and neither of them provides user anonymity. Our scheme removes these weaknesses even with slightly less computation. Results from Tables 2 and 3 shows that our scheme is an efficient improvement of Farash's scheme without any additional computational load.

9 Conclusion

In this paper, we analyzed Farash's protocol for SIP security. We have shown that Farash's protocol is vulnerable to impersonation attack, password guessing attack and temporary session specific information reveal attack. Furthermore, it does not provide user anonymity. Then we proposed an enhanced protocol to overcome security weaknesses of

Table 3 Comparison of efficiency: computational cost/complexity

Scheme→ Security Characteristics↓	Zhang et al.	Tu et al.	Farash	Proposed
Resists user impersonation attack	No	No	No	Yes
Provides user anonymity and user un-traceability	No	No	No	Yes
Resists password guessing attack	No	No	No	Yes
Resists session-specific temporary information attack	Yes	No	No	Yes
Resists replay attack	Yes	Yes	Yes	Yes
Provides mutual authentication	Yes	Yes	Yes	Yes
Resists man-in-the-middle attack	Yes	No	Yes	Yes

Farash's protocol. The proposed protocol is provably secure. We have also analyzed the security of proposed protocol using automated verification tool ProVerif. The proposed protocol is more secure and more efficient as compared with existing protocols.

Acknowledgments This research is supported by the National Natural Science Foundation of China under Grant No. 61300220, and it is also supported by PAPD and CICAEET and Fujian Education and Scientific Research Program for Young and Middle-aged Teachers under Grant No. JA14369. The authors also extend their sincere appreciations to the Deanship of Scientific Research at King Saud University for its funding this Prolific Research Group (PRG-1436-16).

References

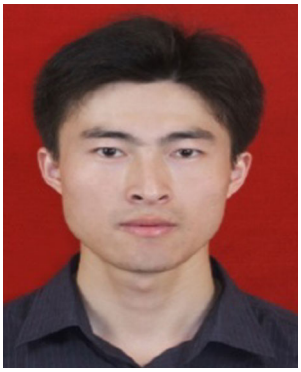
1. Guo P, Wang J, Geng XH, Kim CS, Kim J-U (2014) A variable threshold-value authentication architecture for wireless mesh networks. Journal of Internet Technology 15(6):929–935
2. Farash MS, Chaudhry SA, Heydari M, Sadough S, Mohammad S, Kumari S, Khan MK A lightweight anonymous authentication scheme for consumer roaming in ubiquitous networks with provable security. Int J Commun Syst. doi:10.1002/dac.3019
3. Farash MS, Attari MA (2013) An enhanced authenticated key agreement for session initiation protocol. Information Technology And Control 42(4):333–342
4. ul Amin N, Asad M, Din N, Ashraf Ch S (2012) An authenticated key agreement with rekeying for secured body sensor networks based on hybrid cryptosystem. In: 2012 9th IEEE International Conference on networking, sensing and control (ICNSC). IEEE, pp 118–121
5. Farash MS, Attari MA An anonymous and untraceable password-based authentication scheme for session initiation protocol using smart cards. Int J Commun Syst. doi:10.1002/dac.2848
6. Irshad A, Sher M, Rehman E, Ch SA, Hassan MU, Ghani A (2013) A single round-trip sip authentication scheme for voice over internet protocol using smart card. Multimedia Tools and Applications:1–18
7. Irshad A, Sher M, Faisal MS, Ghani A, Ul Hassan M, Ch SA A secure authentication scheme for session initiation protocol by using ecc on the basis of the tang and liu scheme, Security and Communication Networks
8. Giri D, Srivastava PD (2007) An asymmetric cryptographic key assignment scheme for access control in tree structural hierarchies. IJ Netw Secur 4(3):348–354
9. Islam SH, Khan MK (2014) Provably secure and pairing-free identity-based handover authentication protocol for wireless mobile networks. Int J Commun Syst. n/a–n/a doi:10.1002/dac.2847
10. Islam S, Khan M Cryptanalysis and improvement of authentication and key agreement protocols for telecare medicine information systems. J Med Syst 38(10). doi:10.1007/s10916-014-0135-9
11. Islam S, Biswas G (2011) A more efficient and secure id-based remote mutual authentication with key agreement scheme for mobile devices on elliptic curve cryptosystem. J Syst Softw 84(11):1892–1898
12. Liu J, Zhang Z, Chen X, Kwak KS (2014) Certificateless remote anonymous authentication schemes for wirelessbody area networks. IEEE Transactions on Parallel and Distributed Systems 25(2):332–342
13. Jiang Q, Ma J, Tian Y (2015) Cryptanalysis of smart-card-based password authenticated key agreement protocol for session initiation protocol of zhang et al. Int J Commun Syst 28(7):1340–1351
14. Jiang Q, Ma J, Li G, Yang L (2014) An efficient ticket based authentication protocol with unlinkability for wireless access networks. Wirel Pers Commun 77(2):1489–1506
15. Jiang Q, Ma J, Lu X, Tian Y (2014) An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks. Peer-to-Peer Networking and Applications:1–12. doi:10.1007/s12083-014-0285-z
16. Li X, Niu J, Liao J, Liang W (2015) Cryptanalysis of a dynamic identity-based remote user authentication scheme with verifiable password update. Int J Commun Syst 28(2):374–382. doi:10.1002/dac.2676
17. Kumari S, Khan MK (2014) Cryptanalysis and improvement of a robust smart-card-based remote user password authentication scheme. Int J Commun Syst 27(12):3939–3955. doi:10.1002/dac.2590
18. He D, Wang D (2014) Robust biometrics-based authentication scheme for multiserver environment. IEEE Syst J PP(99):1–8. doi:10.1109/JSYST.2014.2301517
19. He D, Chen J, Chen Y (2012) A secure mutual authentication scheme for session initiation protocol using elliptic curve cryptography. Security and Communication Networks 5(12):1423–1429. doi:10.1002/sec.506
20. Mehmood Z, Nizamuddin N, Ch S, Nasar W, Ghani A (2012) An efficient key agreement with rekeying for secured body sensor networks. In: 2012 2nd international conference on digital information processing and communications (ICDIPC). IEEE, pp 164–167
21. Chaudhry SA, Naqvi H, Shon T, Sher M, Farash M Cryptanalysis and improvement of an improved two factor authentication protocol for telecare medical information systems. J Med Syst 39(6). doi:10.1007/s10916-015-0244-0
22. He D, Kumar N, Chilamkurti N A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks. Inf Sci. doi:10.1016/j.ins.2015.02.010
23. He D, Zeadally S (2015) Authentication protocol for an ambient assisted living system. IEEE Commun Mag 53(1):71–77
24. Amin R, Biswas G (2015) An improved rsa based user authentication and session key agreement protocol usable in tmis. J Med Syst 39(8):1–14
25. Amin R, Biswas G (2015) A secure three-factor user authentication and key agreement protocol for tmis with user anonymity. J Med Syst 39(8):1–19
26. Amin R, Biswas G (2015) A novel user authentication and key agreement protocol for accessing multi-medical server usable in tmis. J Med Syst 39(3):1–17
27. Zhang L, Tang S, Cai Z (2014) Efficient and flexible password authenticated key agreement for voice over internet protocol session initiation protocol using smart card. Int J Commun Syst 27(11):2691–2702. doi:10.1002/dac.2499
28. Tu H, Kumar N, Chilamkurti N, Rho S (2014) An improved authentication protocol for session initiation protocol using smart card. Peer-to-Peer Networking and Applications:1–8. doi:10.1007/s12083-014-0248-4
29. Farash M (2014) Security analysis and enhancements of an improved authentication for session initiation protocol with provable security. Peer-to-Peer Networking and Applications:1–10. doi:10.1007/s12083-014-0315-x
30. Miller VS (1986) Use of elliptic curves in cryptography. In: Advances in Cryptology CRYPTO 85 Proceedings. Springer, pp 417–426
31. Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48(177):203–209

32. (2000) Certicom research standard for efficient cryptography, sec 1,ec cryptography. ver. 1.0, Tech. rep
33. Messerges TS, Dabbish EA, Sloan RH (2002) Examining smart-card security under the threat of power analysis attacks. IEEE Trans Comput 51(5):541–552
34. Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Advances in Cryptology CRYPTO 99. Springer, pp 388–397
35. Canetti R, Krawczyk H (2001) Analysis of key-exchange protocols and their use for building secure channels. In: Advances in Cryptology EUROCRYPT 2001. Springer, pp 453–474
36. Bellare M, Rogaway P (1994) Entity authentication and key distribution. In: CRYPTO 93 Advances in Cryptology. Springer, pp 232–249
37. Bellare M, Rogaway P (1995) Provably secure session key distribution: the three party case. In: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing. ACM, pp 57–66
38. Chaudhry SA, Farash M, Naqvi H, Sher M (2015) A secure and efficient authenticated encryption for electronic payment systems using elliptic curve cryptography. Electron Commer Res:1–27. doi:10.1007/s10660-015-9192-5
39. Chaudhry SA, Farash MS, Naqvi H, Kumari S, Khan MK (2015) An enhanced privacy preserving remote user authentication scheme with provable security. Security and Communication Networks:1–13. doi:10.1002/sec.1299
40. Xie Q, Dong N, Wong DS, Hu B Cryptanalysis and security enhancement of a robust two-factor authentication and key agreement protocol. Int J Commun Syst. doi:10.1002/dac.2858
41. Chaudhry SA, Naqvi H, Sher M, Farash MS, ul Hassan M (2015) An improved and provably secure privacy preserving authentication protocol for SIP, Peer to peer networking and applications. doi:10.1007/s12083-015-0400-9

Saru Kumari is currently an Assistant Professor with the Department of Mathematics, Ch. Charan Singh University, Meerut, 250004, Uttar Pradesh, India. She received Ph.D. degree in Mathematics in 2012 from C.C.S. University, Meerut, Uttar Pradesh, India. She has published 35 papers in international journals and conferences including 22 research publications in SCI indexed journals. She is reviewer of many International journals, including the Journal of Network and Computer Applications (Elsevier), the Journal of Security and Communication Networks (Wiley), Computers and Electrical Engineering (Elsevier), Electronic Commerce Research Journal (Springer), Journal of International Journal of Distributed Sensor Networks (Hindawi) and International Journal of Communication Systems (Wiley). Her current research interests include Information Security, Digital Authentication, Security of Wireless Sensor Networks and Applied Mathematics.

Shehzad Ashraf Chaudhry received his MS Computer Science with distinction, from International Islamic University Islamabad, Pakistan in 2009 and was awarded Gold Medal. Currently he is working as Lecturer and PhD scholar at the Department of Computer Science & Software Engineering, International Islamic University, Islamabad. He authored more than 20 scientific publications published in different international journals and proceedings. His research interests include Lightweight Cryptography, Elliptic/Hyper Elliptic Curve Cryptography, Multimedia Security, MANETs, SIP authentication, IP Multimedia sub-system and Next Generation Networks.

Fan Wu received the Bachelor degree in Computer Science from Shandong University, Jinan, China in 2003, and received Master degree in Computer Software and Theory from Xiamen University, Xiamen, China in 2008. Now he is a lecturer in Xiamen Institute of Technology, Xiamen, China. His current research interests include information security, internet protocols, and network management.



Xiong Li received his masters degree in mathematics and cryptography from Shaanxi Normal University (SNNU) in 2009 and Ph.D. degree in computer science and technology from Beijing University of Posts and Telecommunications (BUPT) in 2012. Dr. Li now is a lecturer of Hunan University of Science and Technology (HNUST). He has published more than 15 referred journal papers. His research interests include cryptography and information security, etc.

Mohammad Sabzinejad Farash received the B.Sc. degree in Electronic Engineering from Shahid Chamran College of Kerman in 2006, and the M.Sc. degree in Communication Engineering from I. Hussein University in 2009. He also received the Ph.D. degree in Cryptographic Mathematics at the Department of Mathematics and Computer Sciences of Tarbiat Moallem University in Iran in 2013. His research interests are Security Protocols and Provable Security Models.

Muhammad Khurram Khan is currently working at the Center of Excellence in Information Assurance, King Saud University, Saudi Arabia. He has edited seven books and proceedings published by Springer-Verlag and IEEE. He has published more than 200 papers in international journals and conferences and he is an inventor of 10 U.S./PCT patents. Dr. Khan is the Editor-in-Chief of a well-reputed journal Telecommunication Systems (Springer). He is also on the editorial boards of several International SCI journals, including the Journal of Network and Computer Applications (Elsevier), Journal of Security and Communication Networks (Wiley), PLOSONE (USA), Computers and Electrical Engineering (Elsevier), Electronic Commerce Research (Springer), Scientific World Journal, Journal of Computing & Informatics, the Journal of Information Hiding and Multimedia Signal Processing (JIHMSP), and the International Journal of Biometrics (Inderscience). Dr. Khurram is one of the organizing chairs of several top-class international conferences and he is also on the program committee of dozens of conferences. He is a recipient of several national and international awards for his research contributions. In addition, he has been granted several national and international funding projects in the field of Cybersecurity. His current research interests include Cybersecurity, biometrics, multimedia security, and digital authentication.