Robust and Efficient Password Authenticated Key Agreement with User Anonymity for Session Initiation Protocol - Based Communications

Liping Zhang, Shanyu Tang[*], *Senior Member IEEE*, Zhihua Cai

**Corresponding author*

Abstract

A suitable key agreement protocol plays an essential role in protecting the communications over open channels among users using Voice over Internet Protocol (VoIP). This paper presents a robust and flexible password authenticated key agreement protocol with user anonymity for Session Initiation Protocol (SIP) used by VoIP communications. Security analysis demonstrates that our protocol enjoys many unique properties, such as user anonymity, no password table, session key agreement, mutual authentication, password updating freely and conveniently revoking lost smartcards etc. Furthermore, our protocol can resist the replay attack, the impersonation attack, the stolen-verifier attack, the man-in-middle attack, the Denning-Sacco attack, and the offline dictionary attack with or without smartcards. Finally, performance analysis shows that our protocol is more suitable for practical application in comparison with other related protocols.

Keywords: *anonymity; mutual authentication; SIP; elliptic curve*

1. Introduction

Voice over Internet Protocol (VoIP)-based communication systems are undergoing rapid development and attract a great deal of attention. Protection of personal communication information is one of the most important issues in VoIP communications over public channels. But the designers of VoIP communications systems mainly focus on a good level of quality of service and pay little attention to security problems. Due to these reasons, the voice packets transmitted between participants in VoIP communications are exposed to the unsecured Internet, which incurs various possible attacks. Long distance calling with Skype is one of the most popular applications of VoIP communications, and VoIP calls are more likely to be threatened by attacks compared with conventional telephone calls with public switched telephone network.

With the widespread applications of VoIP, Session Initial Protocol (SIP) is receiving a lot of attention. SIP was proposed by Internet Engineering Task Force Network Working Group in 1999 [1]. Now, it is widely used as a text-based signalling protocol for VoIP. The architecture of SIP contains five main components: proxy server, redirect server, user agent, register server, and location server.

Compared with other signalling protocols such as H.323, SIP is more flexible and lightweight. But the authentication of SIP is vulnerable to several types of security threats and attacks since it is inherited from HTTP Digest authentication. Most security requirements, such as confidentiality, integrity, authenticity and privacy, can be addressed by building an authentication key agreement protocol. In order to achieve a comparable level of network security with PSTN, an efficient authentication key agreement should be provided to realize the security requirements mentioned above.

The security requirements of an authentication protocol for SIP can be summarized as follows: (1) Resistance to password guessing attacks, stolen-verifier attacks, and server-spoofing attacks; (2) Mutual authentication and session key agreement; (3) Providing user anonymity, i.e. the real identity of the user should be kept anonymity from any third party to protect the privacy of the user; (4) Security in updating passwords; (5) Perfect forward secrecy and known-key security; (6) Resistance to replay attacks, man-in-middle attacks, modification attacks and Denning-Sacco attacks; (7) Resistance to offline dictionary attacks with/without smartcards.

Since original SIP authentication mechanisms could not protect privacy and valuable information over voice communications, several new authentication key agreement protocols have been proposed to provide strong security protection for prevalent VoIP services. As VoIP communication is more sensitive to transmission latency [2], providing a suitable key agreement protocol for SIP should not only meet security demands but also satisfy the requirement of transmission latency. The main objective of our study is to address these problems by constructing a robust and efficient authenticated key agreement protocol that provides strong security protection for SIP without sacrificing the quality of real-time VoIP communication.

The rest of this paper is organized as follows. Section 2 describes related work. Preliminaries are reviewed in Section 3. Section 4 presents our authenticated key agreement protocol. In Section 5, the security of our protocol is discussed in detail. The performance of the protocol is evaluated in Section 6, and the paper is concluded in Section 7.

2. Related Work

Developing a secure user authentication protocol is a critical issue in SIP-based communication services. To date, several protocols have been suggested to seek ways of strengthening the security of SIP authentication process. The original authentication protocol for SIP was based on hyper text transport protocol (HTTP) digest authentication, which offers one-way authentication, but cannot support integrity and confidentiality protection. So the original authentication protocol was not good enough for providing acceptable security level in practice.

Several authenticated key agreement protocols have been proposed in order to strengthen the security of SIP. In 2005, Yang [3] found that the original SIP authentication protocol incurred off-line password guessing attacks and server-spoofing attacks; they constructed a Diffie-Hellman key exchange-based SIP authentication scheme to solve the problems. But their protocol required the server storing a password table for verification purposes and involved expensive exponential computation which is not practical for SIP. Huang [4] later claimed that Yang’s protocol suffered two weaknesses when applied to SIP. One was vulnerable to off-line password-guessing attacks and the other was requiring the execution of expensive exponential operations. So Yang’s protocol was not suitable for low computational power devices. With these points in mind, Huang proposed a new authentication scheme for SIP.

Unfortunately, in [5] Jo discussed the cryptanalysis of Yang’s and Huang’s authentication protocols and demonstrated that both protocols were vulnerable to off-line password guessing attacks. To avoid the requirement of large Public Key Infrastructure (PKI), identity-based cryptography was employed by Ring [6] to construct an authenticated key agreement for SIP. Wang [7] found that Ring’s protocol suffered escrow key problems, and presented a certificateless cryptography-based key agreement protocol, but the computational cost was very high.

In an attempt to improve efficiency, Wu [8] proposed an SIP authentication scheme based on Elliptic Curve Cryptograph (ECC) [19, 20] and proved its security formally by using Canetti-Krawczyk security model. Compared with previous schemes, Wu’s protocol was more efficient, but Liao argued that the distribution of shared secret keys in Wu’s protocol was a great obstacle to scaling up and the protocol failed to take system reparability into considerations [9]. To solve these problems, smartcards were employed to construct a password authenticated key agreement protocol for SIP by using a self-certified public key on elliptic curves. However, Liao’s protocol was susceptible to stolen smartcard attacks. Ni [10] indicated that there was scaling up problems associated with Wu's protocol, and proposed an identity-based authenticated key agreement protocol under Canetti-Krawczyk Model by using elliptic curve cryptography. Furthermore, Yoon [11] demonstrated that Wu's SIP authentication protocol could not resist off-line password guessing attacks, Denning-Sacco attacks and Stolen-verifier attacks; Yoon proposed an authentication scheme to improve the security and to exploit the key block size, speed and security. Unfortunately, both Pu [12] and Gokhroo [13] claimed that Yoon's scheme was vulnerable to off-line password guessing attacks and replay attacks. In 2009, Tsai [14] suggested a nonce-based SIP authentication scheme by using only one-way hash function and exclusive-or operations, thus reducing computational costs efficiently. However, in [15] Yoon argued that Tsai's scheme was vulnerable to off-line password guessing attacks, Denning-Sacco attacks and stolen-verifier attacks. In addition, Tsai's scheme could not provide perfect forward secrecy, and they proposed a new scheme to overcome the weakness. In 2012, Xie [16] demonstrated that Yoon's scheme [15] could not resist stolen-verifier attacks and off-line password guessing attacks. They also proposed an improved protocol based on elliptic curve cryptography to solve above security problems. Moreover, Arshad [17] also indicated some security problems of Tsai's scheme. Besides the two attacks mentioned above, they claimed that Tsai's scheme did not provide known-key secrecy and perfect forward secrecy. In order to eliminate such security flaws, they proposed a mutual authentication protocol for SIP based on elliptic curve discrete logarithm problem. Most recently, He [18] presented a cryptanalysis of Arshad's protocol and found that Arshad's protocol could not resist off-line password-guessing attacks too. Based on Arshad's work, He suggested an improved mutual authentication protocol for SIP, which is immune to several possible attacks.

Nevertheless, the aforementioned protocols have several weaknesses. First, these protocols require SIP server to store a password or verification table consisting of the passwords or hashed passwords of all registered users for verifying the validity of users, making these protocols suffer from some attacks such as password guessing attacks, stolen-verifier attacks and server-spoofing attacks. In addition, since the password or verification tables are usually very large, maintaining these tables makes these solutions hard to scale up, and the reset password problem decreases its applicability for practical use. Second, the aforementioned protocols failed to address the privacy issue of individuality as the real identities of users are transmitted in plaintext, adversaries could using intercepted messages to launch some attacks. Third, the aforementioned protocols cannot provide password-updating functionality for users to change their password as requested.

To attack the problems mentioned above, we propose a robust and flexible password authenticated key agreement with user anonymity for SIP in this study.

3. Preliminaries

We first introduce the basic concepts of the elliptic curve cryptosystem and associated difficult problems, then review original SIP authentication.

3.1 Elliptic curve group and complexity problems

In an elliptic curve cryptosystem, the elliptic curve equation is defined as the form of $E_p(a,b): y^2 = x^3 + ax + b (\bmod p)$ over a prime finite field F_p, where $a,b \in F_p$ and $4a^3 + 27b^2 \neq 0 (\bmod p)$. Given an integer $t \in F_p^*$ and a point $P \in E_p(a,b)$, the scalar multiplication tP over $E_p(a,b)$ can be computed as: $tP = P + P + ... + P$ (t times).

Definition 1. Given two points P and Q over $E_p(a,b)$, the Elliptic Curve Discrete Logarithm Problem (ECDLP) is to find an integer $t \in F_p^*$ such that $Q = tP$.

Definition 2. Given three points P, sP and tP over $E_p(a,b)$ for $s,t \in F_p^*$, the Computational Diffie-Hellman Problem (CDHP) is to find the point stP over $E_p(a,b)$.

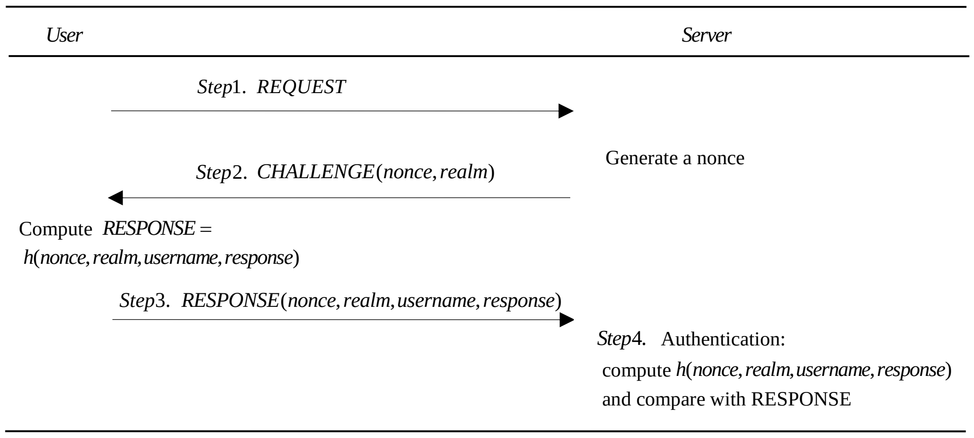
Definition 3. Given two points P and $Q = sP + tP$ over $E_p(a,b)$ for $s,t \in F_p^*$, the Elliptic Curve Factorization Problem (ECFP) is to find two points sP and tP over $E_p(a,b)$.

Definition 4. Given two points P and sP over $E_p(a,b)$ for $s \in F_p^*$, the Inverse Computational Diffie-Hellman Problem (ICDHP) is to find the point $s^{-1}P$ over $E_p(a,b)$.

We assume that the four problems above are intractable. That is, there is no polynomial time algorithm that can solve these problems with non-negligible probability.

3.2 SIP authentication procedure

SIP authentication security depends on the challenge-response mechanism. The original SIP authentication protocol requires the user and the server pre-sharing a password beforehand [1]. This pre-shared password is used to verify the identity of the user or the server in the authentication procedure. The original procedure performs the following steps as shown in Fig. 1.



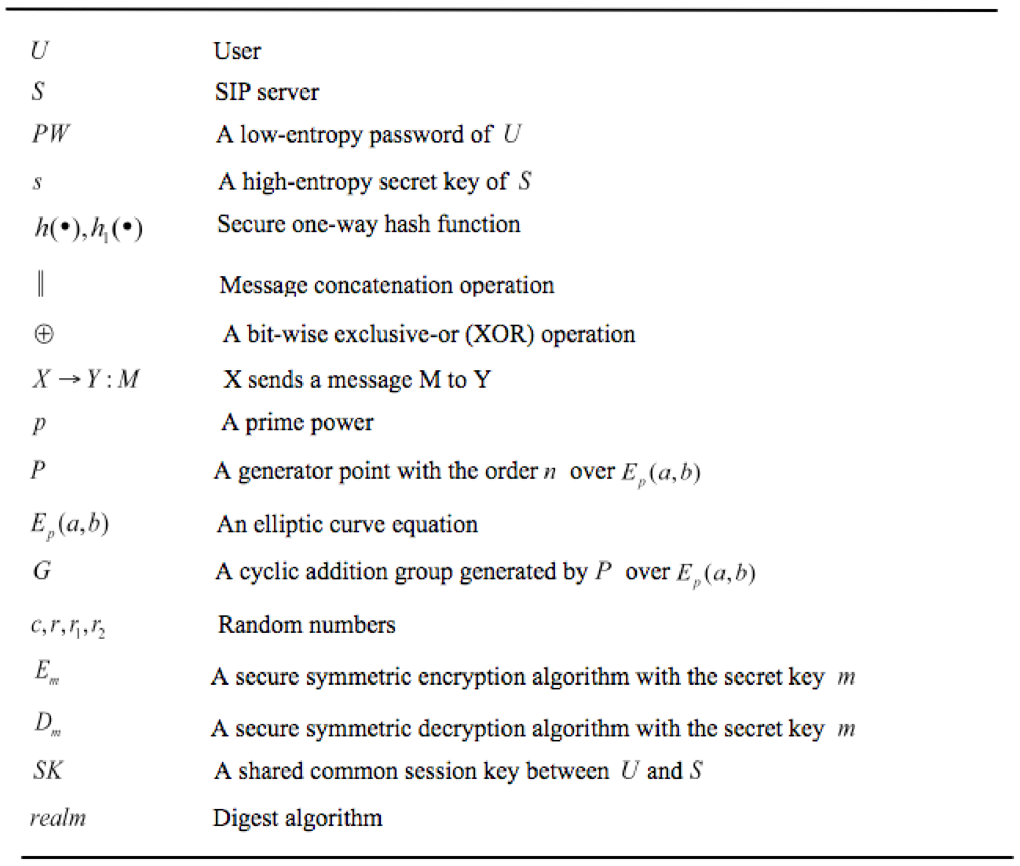
User
Server
Step1. REQUEST
Generate a nonce
Step2. CHALLENGE(nonce, realm)
Compute RESPONSE = h(nonce, realm, username, response)
Step3. RESPONSE(nonce, realm, username, response)
Step4. Authentication: compute h(nonce, realm, username, response) and compare with RESPONSE

Fig. 1 SIP authentication procedure

4. Our Authenticated Key Agreement Scheme

This section presents our newly designed SIP authentication key agreement protocol with user anonymity. In our protocol, there are two entities, the user's smartcard and the server. Our protocol consists of five phases: system setup, registration, pre-computation, authentication and password changing. The notations used in this paper are summarized in Table 1.

Table1. Notations and Terminology



U	User
S	SIP server
PW	A low-entropy password of U
s	A high-entropy secret key of S
$h(\bullet), h_1(\bullet)$	Secure one-way hash function
$\|$	Message concatenation operation
$\oplus$	A bit-wise exclusive-or (XOR) operation
$X \to Y : M$	X sends a message M to Y
p	A prime power
P	A generator point with the order n over $E_p(a,b)$
$E_p(a,b)$	An elliptic curve equation
G	A cyclic addition group generated by P over $E_p(a,b)$
c, r, r_1, r_2	Random numbers
E_m	A secure symmetric encryption algorithm with the secret key m
D_m	A secure symmetric decryption algorithm with the secret key m
SK	A shared common session key between U and S
$realm$	Digest algorithm

4.1. System setup phase

In this phase, the user U and the SIP server S agree on some system parameters.

$StepS1$: Server S first generates a large prime p and an elliptic curve equation $E_p(a,b)$ with order n, which is defined in Section 3.1.

$StepS2$: S finds a generator point P with order n over $E_p(a,b)$, where n is a large number of security considerations. Then, S chooses a random integer $s \in_R Z_p^*$ as a secret key.

$StepS3$: S constructs two secure one-way hash functions $h(\cdot):\{0,1\}^* \rightarrow \{0,1\}^k$, $h_1(\cdot): G \times G \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^k$, where G is a cyclic addition group that is generated by P over $E_p(a,b)$.

$StepS4$: S keeps s secret and publishes information $\{E_p(a,b), P, h(\cdot), h_1(\cdot)\}$.

4.2. Registration phase

When User U wants to register with SIP server S , it performs the following steps with S over a secure channel.

$StepR1$: $U \rightarrow S : (h(PW \| c) \oplus h(username \| c))$

S first verifies U through a secure identification protocol. If U is eligible, it freely chooses its user name $username$, password PW and a random integer $c \in_R Z_p^*$. Then U computes $h(PW \| c) \oplus h(username \| c)$ and submits it to S over a secure channel.

$StepR2$: $S \rightarrow U$: Smartcard contains (R,T)

After receiving information from U, S computes two secrets $R = (h(PW \| c) \oplus h(username \| c))s^2 P$ and $T = E_s(h(PW \| c) \oplus h(username \| c))$ by using its secret key s and the received message $h(PW \| c) \oplus h(username \| c)$. Then S stores the secure information (R,T) in the memory of U's smartcard and issues it to U through a secure channel.

$StepR3$: Smartcard contains (R,T,a)

Upon receiving the smartcard, U writes the secret message c to the memory of the smartcard. Then U's smartcard contains (R,T,c) . Finally, U keeps PW and the smartcard secretly for authentication.

For each user, the registration phase performs once.

4.3. Pre-computation phase

The smartcard generates a random integer $r_1 \in_R Z_p^*$ and computes $W = r_1 R$ and $r_1 P$. At the end, a three-tuple $(W, r_1 P, r_1)$ is stored into the smartcard. Furthermore, the three-tuple is moved from the smartcard after authentication. This means the three-tuples are different in each session.

4.4. Authentication phase

When user U wants to log into SIP server S, it must inserts its smartcard to a card reader and types its user name $username$ and password PW. Moreover, the smartcard has to complete the pre-computation phase. Then the smartcard and the server cooperate to perform the following steps to authenticate each other as shown in Fig. 2.

$StepA1$: $U \to S : REQUEST(W, V)$

U computes $m = (h(PW \| c) \oplus h(username \| c)) r_1 P$ by using its PW, $username$ and the secret message c, r_1 stored in the smartcard. It then uses m to encrypt $r_1 P$ and T as $V = E_m(r_1 P \| h(PW \| c) \oplus h(username \| c) \| T)$. Next, it sends a request message $REQUEST(W, V)$ to S over a public channel.

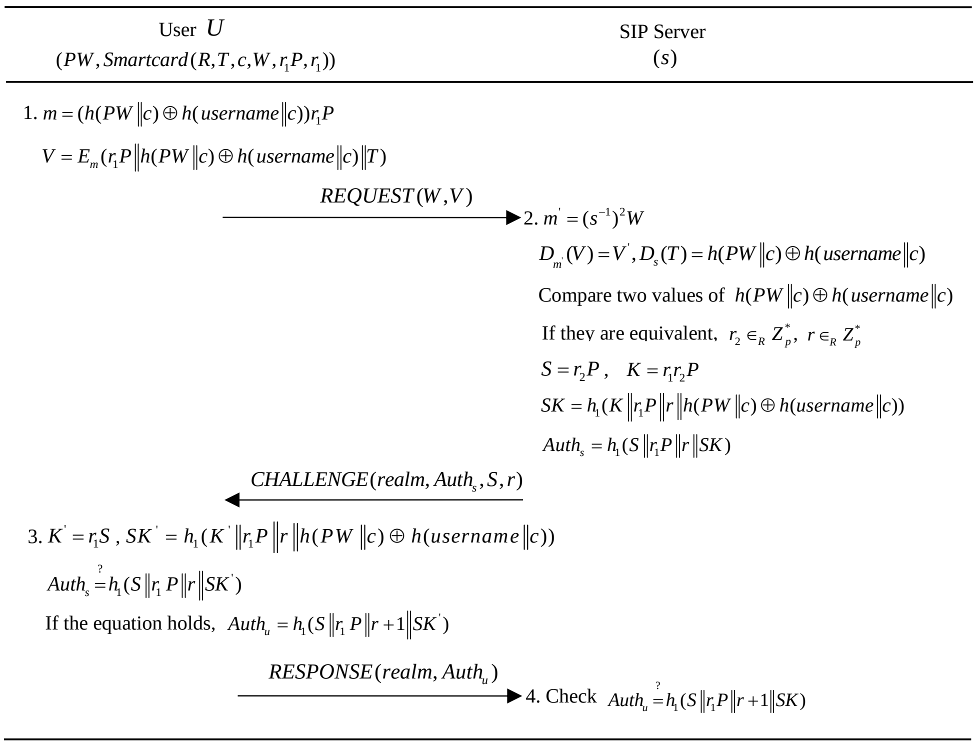
$StepA2$: $S \to U : CHALLENGE(realm, Auth_s, S, r)$

After receiving the request message, S computes $m' = (s^{-1})^2 W$ by using its secret key s. It then decrypts V to get $r_1 P, h(PW \| c) \oplus h(usernamec)$ and T by using m'. Next, S uses its secret key s to decrypt the message T and compares the value of the message $h(PW \| c) \oplus h(username \| c)$ in T with that of the $h(PW \| c) \oplus h(username \| c)$ in V. If they are not equivalent, S rejects the user's request message. Otherwise, S chooses two random integers $r_2 \in_R Z_p^*$ and $r \in_R Z_p^*$, computes $S = r_2 P$, $K = r_1 r_2 P$, the session key $SK = h_1(K \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$ and $Auth_s = h_1(S \| r_1 P \| r \| SK)$ by using the decrypted messages $r_1 P$ and $h(PW \| c) \oplus h(username \| c)$. Finally, it sends $CHALLENGE(realm, Auth_s, S, r)$ to U over a public channel.

$StepA3: U \rightarrow S: RESPONSE(realm, Auth_u)$

After receiving the challenge message, U computes $K^{'} = r_1 S = r_1 r_2 P$ and $SK^{'} = h_1(K^{'} \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$. Then it verifies whether the following equation holds $Auth_s \overset{?}{=} h_1(S \| r_1 P \| r \| SK^{'})$. If held, it computes $Auth_u = h_1(S \| r_1 P \| r+1 \| SK^{'})$ and sends $RESPONSE(realm, Auth_u)$ to S over a public channel. Otherwise, it rejects the challenge message and terminates authentication.

$StepA4$: After receiving the response message, S verifies whether $Auth_u \overset{?}{=} h_1(S \| r_1 P \| r+1 \| SK)$. If the message is authenticated, S sets SK as a shared session key with U; otherwise, it rejects the response message and terminates authentication.



User U
$(PW, Smartcard(R, T, c, W, r_1 P, r_1))$
SIP Server
(s)
1. $m = (h(PW \| c) \oplus h(username \| c)) r_1 P$
$V = E_m(r_1 P \| h(PW \| c) \oplus h(username \| c) \| T)$
$REQUEST(W, V)$
2. $m^{'} = (s^{-1})^2 W$
$D_{m^{'}}(V) = V^{'}, D_s(T) = h(PW \| c) \oplus h(username \| c)$
Compare two values of $h(PW \| c) \oplus h(username \| c)$
If they are equivalent, $r_2 \in_R Z_p^*$, $r \in_R Z_p^*$
$S = r_2 P$, $K = r_1 r_2 P$
$SK = h_1(K \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$
$Auth_s = h_1(S \| r_1 P \| r \| SK)$
$CHALLENGE(realm, Auth_s, S, r)$
3. $K^{'} = r_1 S$, $SK^{'} = h_1(K^{'} \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$
$Auth_s \overset{?}{=} h_1(S \| r_1 P \| r \| SK^{'})$
If the equation holds, $Auth_u = h_1(S \| r_1 P \| r+1 \| SK^{'})$
$RESPONSE(realm, Auth_u)$
4. Check $Auth_u \overset{?}{=} h_1(S \| r_1 P \| r+1 \| SK)$

Fig. 2 Authenticated key agreement phase

4.5. Password changing phase

If user U wants to change its password, it needs to agree on a session key with Server S via the authentication phase in advance. The password-changing steps are executed as follows:

$StepP1$: U encrypts the new password message $(h(PW^* \| c^*) \oplus h(username \| c^*))$ by using the session key SK. Next, it sends $E_{SK}(h(PW^* \| c^*) \oplus h(username \| c^*) \| N \| h(h(PW^* \| c^*) \oplus h(username \| c^*) \| N))$ and N to S, where N is a nonce for freshness checking.

$StepP2$: S decrypts the encryption message and verifies whether the authentication tag $h(h(PW^* \| c^*) \oplus h(username \| c^*) \| N)$ is valid. If not, it rejects the password changing requirement. Otherwise, it computes a new secret value $R^* = (h(PW^* \| c^*) \oplus h(username \| c^*))s^2P$ and $T^* = E_s(h(PW^* \| c^*) \oplus h(username \| c^*))$. It then sends $E_{SK}(R^* \| T^* \| h(N+1 \| R^* \| T^*))$ to U.

$StepP3$: U decrypts the message and verifies whether the authentication tag $h(N+1 \| R^* \| T^*)$ is valid. If so, U stores (R^*, T^*, c^*) in its smartcard.

5. Security Analysis

In this section, we discuss the security of our authentication key agreement protocol for SIP.

5.1. Security properties

The security properties that should be considered for SIP authentication protocols are Replay attacks, Man-in-the-middle attacks, Modification attacks, Denning-Sacco attacks, Stolen-verifier attacks, Offline dictionary attacks without smartcards, Offline dictionary attacks with smartcards, Session key security, Known-key security, Perfect forward secrecy, and Mutual authentication.

5.2. Security analysis

Since the formal security proof of smartcard-based authentication key agreement protocols is still an open problem, all published related protocols have been demonstrated with a simple proof. In this section, we follow the approaches used by Wu [21] and He [18]. First, we discuss the security of our protocol by analyzing some possible attacks, and then evaluate the security of our protocol.

Proposition 1. *Our protocol can resist replay attacks*.

Proof. With the assumption that an adversary Bob intercepts the user $U's$ previous request message $REQUEST(W, V)$ in $StepA1$ and replays it to the SIP server S to impersonate the user U. However, Bob cannot construct a valid $Auth_u$ and send it to the SIP server in $StepA3$ unless he can correctly guess the session key SK' and r_1P. When Bob tries to guess the SK' and r_1P from the intercepted message W, he still faces the elliptic curve discrete logarithm problem. Moreover, Bob cannot obtain r_1P and SK' by decrypting the intercepted message V without the knowledge of secret key s and random integer r_1, where r_1 is a new integer chosen by the user U randomly in each new session. When Bob tries to guess the secret key s and integer r_1 from W, he faces the elliptic curve discrete logarithm problem. So Bob cannot generate a valid $Auth_u$ to pass the verification process of the SIP server in $StepA4$.

On the other hand, suppose Bob intercepts the previous $CHALLENGE(realm, Auth_s, S, r)$ message from the server in $StepA2$ and replays it to impersonate the server S. The user U will find out this attack by checking $Auth_s \overset{?}{=} h_1(S \| r_1 P \| r \| SK)$ since r_1 is assumed to be generated independently and different in each session. So, Bob cannot pass the verification process of the user U in $StepA3$. In this case, no $RESPONSE$ message is sent back to Bob. Therefore, the replay attack cannot succeed in our protocol.

Proposition 2. *Our protocol can resist man-in-the-middle attacks.*

Proof. In order to prevent man-in-middle attacks, the user U and the server S should authenticate each other, which require Our protocol to provide mutual authentication between them. In our protocol, the user U and the server S share a session key SK only after mutual authentication between them. So, an adversary Bob cannot make the independent connections with the user U or the server S and relay messages between them, making them believe that they are communicating directly to each other over a private connection. Only if Bob passes the verification process of the server S, he can establish a session key SK with S, making the server S believe that it is talking to the user U. When Bob attempts to construct a valid authentication message to pass the verification process of the server S, he has to face the elliptic curve discrete logarithm problem. On the other hand, for the same reason Bob cannot impersonate the server S to share a session key with the user U. Thus, Bob cannot launch the man-in-middle attack to cheat either the user U or the server S.

Proposition 3. *Our protocol can resist modification attacks.*

Proof. Suppose an adversary Bob intends to impersonate the user U, he modifies the $REQUEST$ message by constructing $V^{'}, W^{'}$ and sends a fraud $REQUEST$ message to the server S. After receiving the $REQUEST$ message, the server decrypts the $V^{'}$ by using its computed decryption key $m^{'}$. Since Bob does not know the secret key s, the server can find the attack by comparing the value of the message $h(PW \| c) \oplus h(username \| c)$ in T with that of the $h(PW \| c) \oplus h(username \| c)$ in V.

Assuming an adversary Bob chooses $r_2^{'}, r^{'}$ randomly and constructs an authentication message $Auth_s^{'}$ to impersonate the server S and sends a fraud $CHALLENGE(realm, Auth_s^{'}, r_2^{'}P, r^{'})$ to the user U . After receiving the $CHALLENGE$ message, the user U verifies whether the authentication message $Auth_s$ is equivalent to its computed hash value $h_1(S^{'} \| r_1^{'}P \| r \| SK^{'})$. Obviously, Bob cannot go through the verification process of the user U without the knowledge of password PW , username $username$, nonce c and secret message r_1P .

If an adversary Bob attempts to impersonate the user U by modifying the $RESPONSE(realm, Auth_u^{'})$ message and then sending to the server S , where $Auth_u^{'}$ is generated by Bob, for the same reason the server S can easily find the modification by checking $Auth_u^{'} \overset{?}{=} h_1(S \| r_1P \| r+1 \| SK)$. Therefore, our protocol can resist the modification attacks.

Proposition 4. *Our protocol can resist Denning-Sacco attacks.*

Proof. Suppose that an adversary Bob compromises an old session key SK and attempts to find the user's password or other session keys. In our protocol, the session key SK is constructed by $K = r_1r_2P$, r_1P , r and $h(PW \| c) \oplus h(username \| c)$,where the random integers r_1, r_2 and r are different in each session process. When Bob tries to guess the user's password PW or other session keys from the compromised session key SK and other intercepted messages, he not only has to face the elliptic curve discrete logarithm problem, but also needs to break the hash functions, which cannot be solved with non-negligible probability. Therefore, our protocol can protect SIP against Denning-Sacco attacks.

Proposition 5. *Our protocol can resist stolen-verifier attacks.*

Proof. An adversary who steals the password-verifier from the server uses it directly to masquerade as a legitimate user in a user authentication process, which is called the stolen-verifier attack. Our protocol does not maintain any password or verification table on the server. Therefore, the protocol can resist the stolen-verifier attacks.

Proposition 6. *Our protocol can resist offline dictionary attacks without smartcards*.

Proof. Suppose an adversary Bob intends to carry out the offline dictionary attack, and through eavesdropping communications, Bob intercepts all messages relay between the user U and the server S. In order to obtain the user's password PW, Bob needs to extract $h(PW \| c) \oplus h(username \| c)$ from $W = r_1 h(h(PW \| c) \oplus h(username \| c)) s^2 P$, which is equivalent to solving an instance of elliptic curve discrete logarithm problem. So Bob cannot launch the offline dictionary attack by using the $REQUEST$ message. Additionally, when Bob tries to derive PW from the information $Auth_s$ or $Auth_u$, he faces the elliptic curve discrete logarithm problem. Therefore, our protocol can withstand against the offline dictionary attack without smartcards.

Proposition 7. *Our protocol can resist offline dictionary attacks with smartcards.*

Proof. Suppose an adversary Bob compromises the secret information (R, T, c) stored in the smartcard of the user U and intercepts the $REQUEST$ message, the $CHALLENGE$ message and $RESPONSE$ message transmitted between the user U and the server S. Compared with the offline dictionary attack without smartcards, the addition information known by Bob in this attack is (R, T, c). However, Bob cannot extract $h(PW \| c) \oplus h(username \| c)$ from R and check whether each of their guessed passwords is correct or not. Because computing $h(PW \| c) \oplus h(username \| c)$ from R is equivalent to solving an instance of elliptic curve discrete logarithm problem. Furthermore, Bob cannot obtain $h(PW \| c) \oplus h(username \| c)$ from T without the knowledge of the server's secret key s. Therefore, our protocol can resist the offline dictionary attack with smartcards.

Proposition 8. *Our protocol can provide session key security.*

Proof. In our protocol, at the end of the key exchange, the session key $SK = h_1(K \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$ is not known by anyone but only the user U and the server S, since $K = r_1 r_2 P$ and $h(PW \| c) \oplus h(username \| c)$ cannot be constructed correctly by the adversary Bob without the knowledge of $(r_1, c, PW, username)$ or the secret key s and the random integer r_2. So, none of this session key $SK = h_1(K \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$ is known to anybody but the user U and the server S. Therefore, our protocol provides session key security.

Proposition 9. *Our protocol can provide known-key security.*

Proof. In our protocol, the random numbers r_1 and r_2 generated independently by the server S and the user $U's$ smartcard are different in each session process. So the session key $SK = h_1(K \| r_1 P \| r \| h(PW \| c) \oplus h(username \| c))$ of each session is not connected with the session keys of any other sessions. Even if the adversary Bob compromises a session key SK and the random values r_1 and r_2, he cannot compute other session keys $SK = h_1(r_1^{'} r_2^{'} P \| r_1^{'} P \| r^{'} \| h(PW \| c) \oplus h(username \| c))$. This is because in each session a fresh session key is generated depending on $(r_1^{'} r_2^{'} P, r_1^{'} P, r^{'})$, and the secret differs in every session. Therefore, in our protocol, each run of the authentication and key agreement process produces a unique session key SK between the user U and the server S.

Proposition 10. *Our protocol can provide perfect forward secrecy.*

Proof. In our protocol, suppose the user's password PW and the server's secret key s are compromised. The adversary Bob may obtain the correct decryption key m by using the secret key s and the intercepted $REQUEST(W,V)$ message. He could then use m to decrypt the message V to get the secret information $\mathrm{r}_1 P$, $h(PW \| c) \oplus h(username \| c)$ and decrypt T to obtain $h(PW \| c) \oplus h(\; username \| c)$. But knowing above information is not enough for computing a previous session key $SK = h_1(r_1^{'} r_2^{'} P \| r_1^{'} P \| r^{'} \| h(PW \| c) \oplus h(username \| c))$, because Bob cannot compute a correct $K = r_1^{'} r_2^{'} P$. In order to obtain K Bob needs to extract the value $r_2^{'}$ from $S = r_2^{'} P$ or the value $r_1^{'}$ from $r_1^{'} P$, which is equivalent to solving an instance of elliptic curve discrete logarithm problem. Therefore, in our protocol, even if the user's password PW and the server's secret key s are compromised, the secrecy of previous session keys established by them cannot be affected.

Proposition 11. *Our protocol can provide mutual authentication.*

Proof. In our protocol, the server S and the user U authenticate each other by checking $Auth_u$ and $Auth_s$, respectively. Therefore, our protocol can provide mutual authentication.

Proposition 12. *Our protocol can provide security in choosing and updating passwords.*

Proof. In our protocol, the legitimate user with smartcards can freely choose her or his favourite password in the registration phase. It makes users easy to remember their own passwords. Our protocol also provides an update password phase for users to change their password as requested. Even having been stolen or lost smartcards, any other person cannot change or update the password without knowing the current session key SK shared between the user U and the server S .

Proposition 13. *Our protocol can provide user anonymity*.

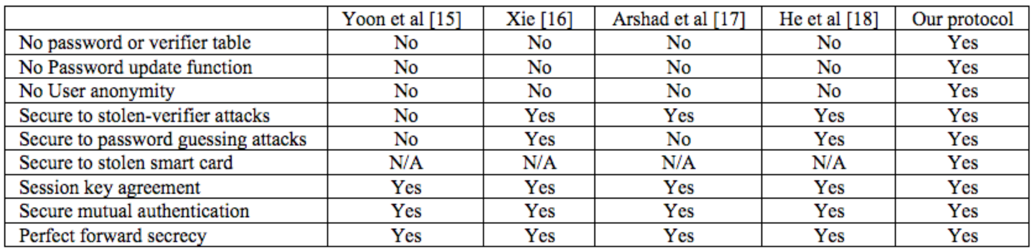
Proof. In our protocol, the anonymity of the user U is obtained by hash function, symmetric encryption technique and elliptic curve discrete logarithm problem. In the registration phase, the user name is protected by the password PW and a nonce c chosen by the user. The hashed user's identity consisting of the username $username$, the password PW and the nonce c is submitted to the server S. Even the server S does not know the real username of the user U. In the authentication phase, the hashed username is protected by a secure symmetric encryption algorithm and elliptic curve discrete logarithm problem. So, even if an adversary Bob compromises the secret (R, T, c) stored in smartcards and record the used messages transmitted between the user U and the server S, he cannot derive the real username of the user without the knowledge of secret key s and user U's password PW.

6. Performance Comparison

In this section, we first summarize the functionality of our protocol and then compare our protocol with relevant protocols in terms of computational cost.

In our protocol, the user's password is embedded in $h(PW\|c) \oplus h(username\|c)$. In the registration phase, after receiving the secret message $\{h(PW\|c) \oplus h(username\|c)\}$, the server computes two secrets $R = (h(PW\|c) \oplus h(username\|c))s^2P$ and $T = E_s(h(PW\|c) \oplus h(username\|c))$, the server stores the secrets in the memory of the user's smartcard, and then delivers the smartcard to the user via a secure channel. Obviously, during the registration process, the server does not need to store any password table. Furthermore, in our protocol, the user's real username is protected by hash function, symmetric encryption technique and elliptic curve discrete logarithm problem. The adversary cannot obtain the real username even if she or he intercepts all the messages transmitted between the user and the server. Therefore, our protocol provides user anonymity to protect the user's privacy. Finally, our protocol provides a secure updating password phase for users to change their password freely and can resist stolen smartcard attacks. As shown in Table 2, our protocol can provide more unique features such as no password or verifier table, user anonymity and password update freely, which has not been considered by other related protocols. In fact, these new features are very important in implementing a practical and universal authenticated key agreement for SIP-based VoIP communications.

Table 2. Functionality comparisons between our protocol and others



	Yoon et al [15]	Xie [16]	Arshad et al [17]	He et al [18]	Our protocol
No password or verifier table	No	No	No	No	Yes
No Password update function	No	No	No	No	Yes
No User anonymity	No	No	No	No	Yes
Secure to stolen-verifier attacks	No	Yes	Yes	Yes	Yes
Secure to password guessing attacks	No	Yes	No	Yes	Yes
Secure to stolen smart card	N/A	N/A	N/A	N/A	Yes
Session key agreement	Yes	Yes	Yes	Yes	Yes
Secure mutual authentication	Yes	Yes	Yes	Yes	Yes
Perfect forward secrecy	Yes	Yes	Yes	Yes	Yes

For the convenience of evaluating computational cost, several notations are defined:

(1) T_{ecsm} is the time for executing a scalar multiplication operation of elliptic curve;

(2) T_{ecpa} is the time for executing a point addition operation of elliptic curve;

(3) T_h is the time for executing a one-way hash function;

(4) T_{inv} is the time for executing a modular inversion operation;

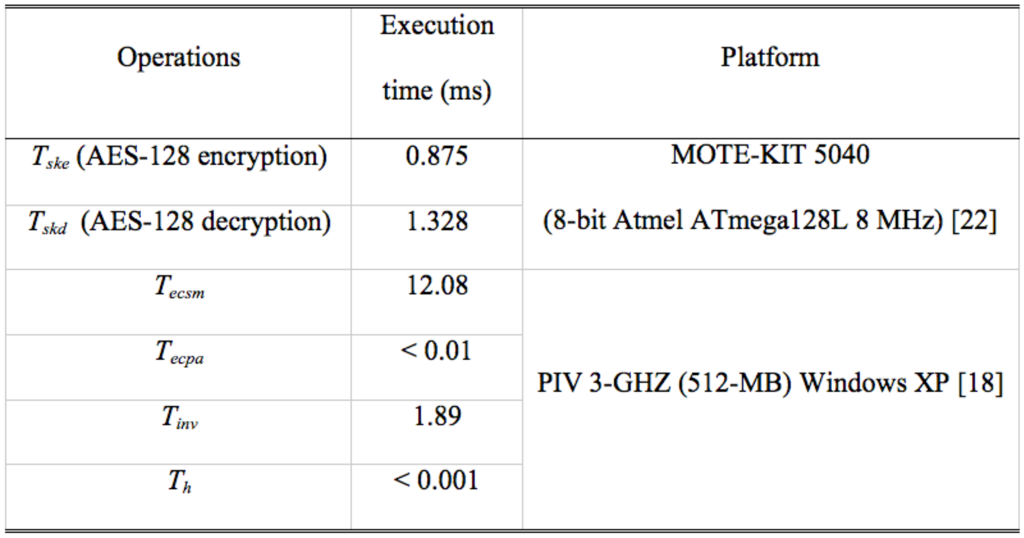
(5) T_{ske} is the time for executing a symmetric key encryption operation;

(6) T_{skd} is the time for executing a symmetric key decryption operation.

Next, we discuss the implementation data. For the convenience of performance analysis, assume the AES cryptosystem is the symmetric encryption/decryption operation in our scheme. Duh [22] implemented AES based on MOTE-KIT 5040 (8-bit Atmel ATmega128L 8 MHz). Their implementation encrypted and decrypted a 128-bit block of plaintext in 0.857 ms (T_{ske}) and 1.328 ms (T_{skd}), respectively. Furthermore, based on PIV 3-GHZ processor with

512-MB memory and a Windows XP operation system, He [18] described the implementation of a scalar multiplication operation of elliptic curve in 12.08ms (T_{ecsm}), and a point addition operation of elliptic curve less than 0.01ms (T_{ecpa}) respectively (ECC group on Koblitz elliptic curve $y^2 = x^3 + ax^2 + b$ defined on $F_{2^{163}}$ with $a = 1$ and b a 163-bit random prime). They also showed that the processor could perform a modular inversion operation in 1.89 ms (T_{inv}) and a one-way hash function operation less than 0.001ms (T_h). The execution times of related operations are summarized in Table 3.

Table 3. Execution times of related operations



Operations	Execution time (ms)	Platform
T_{ske} (AES-128 encryption)	0.875	MOTE-KIT 5040 (8-bit Atmel ATmega128L 8 MHz) [22]
T_{skd} (AES-128 decryption)	1.328	
T_{ecsm}	12.08	PIV 3-GHZ (512-MB) Windows XP [18]
T_{ecpa}	< 0.01	
T_{inv}	1.89	
T_h	< 0.001	

In the registration phase, our protocol requires two hash operations to compute $h(username\|c) \oplus h(PW\|c)$ on the user side, one scalar multiplication operation of elliptic curve to obtain R and one symmetric key encryption operation to get T on the server side. In the pre-computation phase, the smartcard takes two scalar multiplication operations of elliptic curve to obtain W and r_1P. In the authentication phase, the user takes two scalar multiplication operations of elliptic curve to compute $m = (h(PW\|c) \oplus h(username\|c))r_1 P$ and $K^{'} = r_1S$; two symmetric key encryption operations to compute V; and five one-way hash function operations to compute $h(username\|c), h(PW\|c), Auth_s, Auth_u$ and $SK^{'}$. The server takes one scalar multiplication operation of elliptic curve and one modular inversion operation to get $m^{'}$; two scalar multiplication operations of elliptic curve to compute S and K; two symmetric key decryption operation to decrypt V and T; and three one-way hash function operations to compute $SK, Auth_s$ and $Auth_u$. In the password changing phase, the user takes four one-way hash function operations to compute $h(PW^*\|c^*), h(username\|c^*), h(h(PW^*\|c^*) \oplus h(username\|c^*)\|N)$ and $h(N+1\|R^*\|T^*)$; one symmetric key encryption operation and one symmetric key decryption operation. The server takes one scalar multiplication operation of elliptic curve to compute a new secret value R^*; two one-way hash function operations to compute $h(h(PW^*\|c^*) \oplus h(username\|c^*)\|N)$ and $h(N+1\|R^*\|T^*)$; and two symmetric key encryption operation and one symmetric key decryption operation.

Table 4. Computational comparisons between our protocol and others

Phase	Yoon [15]	Xie [16]	Arshad [17]	He [18]	Our protocol
Registration	$2T_{ecsm} + 1T_{ecpa}$ 24.17ms	$1T_{ske}$ 0.875 ms	N/A	$2T_h$ 0.002 ms	$1T_{ecsm} + 2T_h + 1T_{ske}$ 12.975 ms
Pre-computation	N/A	N/A	N/A	N/A	$2T_{ecsm}$ 24.16 ms
Authentication	$6T_{escn} + 3T_{ecpa} + 4T_h$ 72.52 ms	$6T_{ecsm} + 1T_{ecpa} + 6T_h$ $+1T_{ske} + 1T_{skd} + 1T_{inv}$ 76.57 ms	$5T_{ecsm} + 6T_h + 1T_{inv}$ 62.30 ms	$6T_{ecsm} + 6T_h$ 72.49 ms	$5T_{ecsm} + 8T_h + 1T_{ske}$ $+2T_{skd} + 1T_{inv}$ 65.82 ms
Password changing	N/A	N/A	N/A	N/A	$1T_{ecsm} + 6T_h +$ $3T_{ske} + 2T_{skd}$ 17.376 ms

Table 4 lists computational comparisons between our protocol and other related protocols. According to Tables 3 and 4, the authentication phase of our protocol is estimated to be implemented in 65.82 ms, the times for implementing the authentication phase of Yoon's protocol [15], Xie's protocol [16], Arshad's protocol [17] and He's protocol [18] are 72.52ms, 76.57ms, 62.30 ms and 72.49 ms.

Since Arshad's protocol doesn't provide registration, there is no computational cost for the registration phase. Unlike our protocol, other protocols don't contain the Pre-computation and Password changing phases, thereby suffering from several attacks, so no computational costs are presented in Table 4 for these two phases.

In the pre-computation phase, since the integer r_1 is chosen by smartcard instead of user, the message $W=r_1R$ and r_1P are computed by smartcard at any time before the authentication phase. Although two scalar multiplication operations of elliptic curve are required in the pre-computation phase, it would not affect the performance of the real-time VoIP communication since no data are transmitted during the pre-computation phase. Other protocols without smartcards cannot execute pre-computation, since users choose r_1 differently during each authentication. So the performance is mainly dependent on the authentication phase.

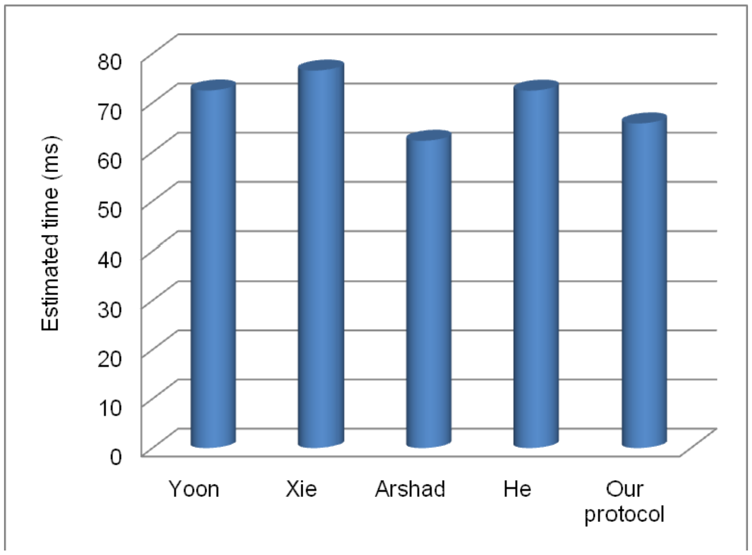


Fig. 3 Execution time comparisons between our protocol and others

As the authentication phase determines overall performance, execution time comparisons for authentication between our protocol and others are shown in Figure 3. Clearly our protocol is as efficient as Arshad's, but is much efficient than the other three protocols (*i.e.* Yoon, Xie, He) in the authentication phase, this is because our protocol reduces the number of operations of scalar point multiplication of elliptic curve.

The total computational overhead of our protocol costs little more, that is because our protocol possesses many unique features, such as not maintaining any password or verification table on the SIP server, users being able to choose or change its own password freely and providing user anonymity to keep user's identity secret while still allowing the server to perform its own authentication, which requires more operations. Moreover, for each user the registration phase performs only once and the password change phase only performs when users need to update their passwords. Therefore, this minor computational increase is indispensable for constructing a reliable and trustworthy authenticated key agreement for SIP used by VoIP-based real-time communications systems.

7. Conclusion

A new password authenticated key agreement protocol with user anonymity for SIP has been proposed in this paper. In our protocol, the user and the server can achieve mutual authentication and key agreement by using passwords and smartcards. Meanwhile, our protocol can withstand replay attacks, impersonation attacks, stolen-verifier attacks, man-in-middle attacks, Denning-Sacco attacks and offline dictionary attacks with or without smartcard. In addition, our protocol also provides some unique features such as user anonymity, no password table needed, revoking lost smartcard conveniently and password updating freely. These new features have not been considered in other related work, but they are very important to implementing a practical and universal authenticated key agreement for SIP.

Acknowledgment

This work was supported by the National Natural Science Foundation of China [Grant numbers 61272469, 61303237, 61075063], Wuhan Scientific Research Program [Grant number 2013010501010144] and China Postdoctoral Fund [Grant number 2012194091].

References

1. Rosenberg J., Schulzrinne H., Camarillo G.: ‘SIP: Session Initiation Protocol’, RFC 3261, June 2002

2. Wang, Chia-Hui, Liu, Yu-Shun: ‘A dependable privacy protection for end-to-end VoIP via Elliptic-Curve Diffie-Hellan and dynamic key changes’, *Journal of Network and Computer Applications*, 2011, **34**, pp.1545-1556

3. Yang, C., Wang, R., Liu, W.: ‘Secure authentication scheme for session initiation protocol’, *Computers & Security*, 2005, **24**, pp.381-386

4. Huang, H., Wei, W., Brown, G.: ‘A new efficient authentication scheme for session initiation protocol’. Proceedings of JCIS 06, Kaohsiung, Taiwan, ROC, October 2006

5. Jo, H., Lee, Y., Kim, M., Kim, S., Won, D.: ‘Off-line Password-Guessing Attack to Yang’s and Huang’s Authentication Schemes for Session Initiation Protocol’. Proceedings of INC, IMS and IDC, Seoul, Korea, August 2009, pp.618-621

6. Ring, J., Choo, K.-K. R., Foo, E., Looi, M.: ‘A new authentication mechanism and key agreement protocol for sip using identity-based cryptography’. Proceedings of AusCERT R&D Stream, Gold Coast, Australia, May 2006, pp.61-72

7. Wang, Fengjiao, Zhang, Yuquing: ‘A new provable secure authentication and key agreement mechanism for SIP using cerificate less public key cryptography’, *Computer Communications*, 2008, **31**, pp.2142-2149

8. Wu, L., Zhang, Y., Wang, F.: ‘A new provably secure authentication and key agreement protocol for SIP using ECC’, *Computer Standards & Interfaces*, 2009, **31**, pp.286-291

9. Liao, Yi-Pin, Wang, Shuenn-Shyang: ‘A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves’, *Computer Communications*, 2010, **33**, pp.372-380

10. Ni, Liang, Chen, Gongligang, Li, Jianhua: ‘A Pairing-free Identity-based Authenticated Key Agreement Mechanism for SIP’. Proceedings of ICNCIS2011, Guilin China, May 2011, pp.209-217

11. Yoon, E.J., Yoo, K.Y.: ‘A secure and efficient SIP authentication scheme for converged VoIP networks’, *Computer Communications*, 2010, **33**, pp.1674-1681

12. Pu, Q.: ‘Weaknesses of SIP authentication scheme for converged VoIP networks’, http://eprint.iacr.org/2010/464

13. Gokhroo, M.K., Jaidhar, C.D., Tomar, A.S.: ‘Cryptanalysis of SIP Secure and Efficient Authentication Scheme’. Proceedings of ICCSN 2011, Xian, China, May 2011, pp.308-310

14. Tsai, Jia Lun: ‘Efficient Nonce-based authentication scheme for session initiation protocol’, *International Journal of Network Security*, 2009, 9, pp.12-16

15.Yoon, E, Shin, Y, Jeon, I, Yoo, K.: ‘Robust mutual authentication with a key agreement scheme for the session initiation protocol’, *IETE Technical Review*, 2010, **27**, pp.203-213

16. Xie, Qi: ‘A new authenticated key agreement for session initiation protocol’, *International Journal of Communication Systems*, 2012, **25**, pp.47-54

17. Arshad, R, Ikram, N.: ‘Elliptic curve cryptography based mutual authentication scheme for session initiation protocol’, *Multimedia Tools and Applications*, 2011, DOI: 10.1007/s11042-011-0787-0

18. He, Debiao, Chen, Jianhua, Chen, Yitao: ‘A secure mutual authentication scheme for session initiation protocol using elliptic curve cryptography’, *Security and Communication Networks*, 2012, DOI:10.1002/sec.506

19. Koblitz, N.: ‘Elliptic curve cryptosystems’, *Mathematics of Computation*, 1978, **48**, pp.203-209

20. Branovic, I., Giorgi, R., Martinelli, E.: ‘A workload characterization of elliptic curve cryptography methods in embedded environment’, *ACM SIGARCH Computer Architecture News*, 2004, **32**, pp.27-34

21. Wu, C.C., Lee, W.B., Tsaur, W.J.: ‘A secure authentication scheme with anonymity for wireless communications’, *IEEE Communication Letter*, 2008, **12**(10) , pp.722-723

22. Duh, D.R., Lin, T.C., Tung, C.H., Chan, S.J.: ‘An implementation of AES algorithm with the multiple spaces random key pre-distribution scheme on MOTE-KIT 5040’. Proc. of IEEE International Conference of Sensor Networks, Ubiquitous, and Trustworthy Computing, Taichun, Taiwan, ROC, June 2006, pp.64-71