A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves

Yi-Pin Liaoa,b,*, Shuenn-Shyang Wanga

a Institute of Communication Engineering, Tatung University, 40, Sec. 3, Chungshan N. Road, Taipei, Taiwan, ROC
b Department of Computer Science and Information Engineering, St. John’s University, 499, Sec. 4, Tam King Road Tamsui, Taipei, Taiwan, ROC

Abstract

Voice over Internet Protocol (VoIP) has received much attention and has become a real competitor to traditional Public Switched Telephone Networks (PSTNs), where the Session Initial Protocol (SIP) is widely used as a signaling protocol based on HTTP-like request/response exchange to establish multimedia sessions in both wireline and wireless world. However, the original authentication scheme for SIP-based service typically uses HTTP Digest authentication protocol, which is not providing security at an acceptable level. In this paper, we present a new secure password authenticated key agreement scheme for SIP-based service using self-certified public keys (SCPKs) on elliptic curves. Due to using SCPKs on elliptic curve, the proposed scheme not only avoids the requirement of a large Public Key Infrastructure (PKI) but also achieve efficient performance in contrast to other public key cryptosystems. The main merits include: (1) it achieves mutual authentication and session key agreement; (2) it does not maintain any password or verification table in the server; (3) it prevents various possible attacks induced by open networks and the standard of SIP message; (4) it can be applied to authenticate the users with different SIP domains; (5) it provides the users to update password quickly and securely; and (6) it can avoid key escrow problem.

1. Introduction

Within the traditional Public Switched Telephone Networks (PSTNs) a good level of quality of service (QoS) and security has been established over the years, and it is now widely guaranteed. With the rapid growth of Internet technology, Voice over Internet Protocol (VoIP) is receiving much attention and becomes a real competitor to traditional PSTN. If VoIP wants to replace PSTN, it should provide the same basic telephone with a comparable level of QoS and network security in many service scenarios. While the problem of QoS mainly concerns IP network layer, the problem of security involves the control architecture and its signal protocol. Among many dedicated protocols used to handle sessions for VoIP, the Session Initial Protocol (SIP) is the widely used [1,2]. SIP is an application layer signaling protocol based on HTTP-like request/response exchange for initiating, managing and terminate voice and video session across packet networks.

1.1. Related works for SIP authentication schemes

The identity authentication is an important issue in SIP-based service. For example, when the user Alice wants to make a SIP voice call to the user Bob, how can he verify that he is connected exactly to SIP user agent of Bob, and not to other client pretending to be the SIP user agent of Bob. However, SIP authentication scheme typically uses HTTP Digest authentication protocol noted in RFC2617 [3] and is not providing security at an acceptable level [4–7]. Although S/MIME can provide SIP message both integrity and confidentiality [8], the usage of S/MIME depending on the existence of the user’s certificates are seriously limited in that there is virtually no consolidated authority today that provides certificates for users applications on a global scale [1]. Moreover, SIP over SSL (SIPS) can also provide end-to-end protection on SIP request/response message. But it still requires end user’s certificate in place and increase the workload of SIP proxy servers significantly.

To guarantee the security of the growing SIP-based services, several new schemes have been proposed to enhance the security of SIP [9–13]. Yang et al. [9] pointed out that HTTP Digest authentication protocol is subject to the off-line guessing attack and the spoofing attack. Then, they introduced a public key cryptosystem based on Diffie–Hellman key exchange protocol to solve these problems. However, Yang et al.’s scheme incurs the replay attack.
Furthermore, it needs to maintain preconfigured password table and involves in exponential computation, which is not suitable for the user’s device with limited computing capability. Recently, Ring et al. [10] provided a secure authenticated key agreement (AK) protocol for SIP using identity-based cryptography (IBC) [14]. It computes the hash value of user’s SIP identity as his public key without the need of concrete certificates. However, Ring et al.’s scheme suffers from the heavy computation load due to costly bilinear pairing and identity-based signatures [14,15]. Additionally, since the trust authority (TA) knows any eligible user’s long-term private key and can therefore impersonate any user without being detected. On the other hand, the escrow key problem can be caused by way of the collusion with the relevant TAs [16]. To solve those problems of Ring et al.’s scheme, Wang and Zhang [11] proposed a new secure authentication and key agreement (SAKA) mechanism using certificateless public key cryptography (CL-PKC) [17]. Wang and Zhang’s scheme emphasizes that TA cooperates with the communication entity to generate the private key. Thus, Wang and Zhang’s scheme can avoid key escrow problem while maintaining the heavy computation load unsolved. At the same time, Geneiatakis and Lambrinoudakis (2008) [12] proposed an improved authentication scheme to enhance the security of HTTP Digest authentication for SIP. They introduce a new SIP header, namely the Integrity-Auth header, which is aiming at protecting the SIP-based services from signaling attacks while ensuring authenticity and integrity. However, the Integrity-Auth header involves the hash value of the user’s password combined with some known parameters. Under this situation, the password table or verifier table still be maintained in the servers and susceptible to stolen-verifier table attack. In addition, the offline password guessing attack cannot be avoided. Lately, Wu et al. [13] also presented a new authenticated key exchange protocol NAKE to solve the existing problems in SIP original authentication. Wu et al.’s scheme assumes that the communication parties must share a common secret $k$ beforehand between the ISIM (i.e., smart card-like device) and the Authentication Center (AuC). Once the secret key $k$ is leaked for some reasons, the adversary can easily launch the forgery attack to masquerade as the user client or the server. Although pre-shared key (PSE) is the most cost effective way but the problem of distributing the shared secrets makes this solution hard to scale. Furthermore, Wu et al.’s scheme does not take the system reparation into considerations [18].

In this paper, we propose a new secure SIP authentication scheme using self-certified public keys (SCPKs). Our proposed scheme emphasizes that it does not only solve the problems caused by related works but also migrates the computation overhead in contrast to other public key cryptosystems. Additionally, the main merits include: (1) it achieves mutual authentication and session key agreement; (2) it does not maintain any password or verification table in the server; (3) it prevents various possible attacks induced by open networks and the standard of SIP message; (4) it can be applied to authenticate the users with different SIP domains; (5) it provides the users to update password quickly and securely; and (6) it can avoid key escrow problem.

The remainder of the paper is organized as follows. In Section 2, HTTP Digest authentication scheme for SIP is introduced, including the authentication procedure and the security issues retained for SIP-based service. Then, we briefly review the basic concepts on self-certified public keys (SCPKs) and some related mathematical problems in Section 3. Section 4 presents a new secure authentication scheme for SIP. Section 5 shows the correctness and security analysis. In Section 6, we evaluate the performance and compare the functionality between the proposed scheme and the others. Finally, the conclusion is given in Section 7.

2. Review of HTTP Digest authentication scheme for SIP

2.1. SIP protocol overview

A VoIP infrastructure inherits and utilizes various protocols from the Internet stack architecture. Specifically, SIP is an application-layer signaling protocol for creating, modifying and terminating multimedia sessions among one or more participants. The network entities involved in SIP are composed of user agent, proxy servers, redirect servers and registrar servers, which are depicted in Fig. 1. The user agents represent the terminal (i.e., the user agent client (UAC) Alice and user agent server (UAS) Bob). The proxy server is an intermediary entity that as both a server and a client for making requests on behalf of other clients (i.e., the proxy servers in SIP domain A and domain B). The redirect server accepts requests and replies to the client with a response message (typically providing a contract address for the called user). The registrar is a particular that accepts user registration requests (i.e., the registrars in SIP domain A and domain B). SIP signal between two user agents consists of requests and response. Before Alice initiates a call, she registers its current address to registrar server by way of sending the REGISTER message, and then store address bindings in location server. A simple call setup with proxy between two user agents is shown in Fig. 2, where Alice represents the user agent client (UAC) that acts as the initiator for a call (i.e., caller), and the user agent server (UAS) Bob as the terminator for a call (i.e., callee). A successful SIP invitation consists of two requests, the INVITE message followed by an ACK message. First, Alice sends an INVITE message to locally configured proxy server. After receiving the INVITE message, the proxy server corresponding to Alice will transmit the INVITE message to the proxy server corresponding to Bob by way of making a DNS lookup through redirect server. Then the proxy server corresponding to Bob query the location server to retrieve the IP address of Bob, and forwards the INVITE message to Bob. When Bob accepts the call, both the informational RING and the OK messages are issued, and takes return path via the proxy server hops. Thereafter, Alice sends an ACK message straight to Bob, without proxies, and the whole SIP conversation continues peer-to-peer. When the caller or callee wishes to terminate a call, either of them sends a BYE request. The mechanisms that provide security in SIP can be classified as end-to-end or hop-by-hop protection. However, SIP authentication scheme is applied only to end-to-end (i.e., caller to callee) communication; hop-to-hop (i.e., proxy to proxy) authentication should rely on other mechanism like IPSec [19] or TLS [20].

![Fig. 1. SIP-based infrastructure between two SIP domains.](image-url)
Step 4: According to the username, the server extracts the user credentials. The authentication procedures are shown in Fig. 3 and described as follows:

- **Step 1**: The user client sends a REQUEST message to one server.
- **Step 2**: The server responds a CHALLENGE message to the user client, including a nonce value and realm.
- **Step 3**: After receiving the CHALLENGE message, the user client utilizes the secret password shared with the server, together with the received nonce value and the username, to compute a response as following equation:
  \[ \text{response} = F(\text{nonce}, \text{username}, \text{password}, \text{realm}) \]  
  Then, the user client sends the REQUEST message again with the computed response value, username, nonce value and realm.
- **Step 4**: According to the username, the server extracts the user client’s password. Then the server verifies whether the nonce is correct. If it is correct, the server computes as Eq. (1) and checks whether the computed value is equal to the received response value. If they match, the server authenticates the identity of the client.

2.2. HTTP Digest authentication protocol

SIP presently uses HTTP Digest authentication protocol, with an option to use certificates and a PKI. HTTP Digestion authentication protocol is mainly based on challenge/request message exchange to verify the SIP entities, including user agents, registrars and proxies. The authentication procedures are shown in Fig. 3 and described as follows:

- **Step 1**: The user client sends a REQUEST message to one server.
- **Step 2**: The server responds a CHALLENGE message to the user client, including a nonce value and realm.
- **Step 3**: After receiving the CHALLENGE message, the user client utilizes the secret password shared with the server, together with the received nonce value and the username, to compute a response as following equation:
  \[ \text{response} = F(\text{nonce}, \text{username}, \text{password}, \text{realm}) \]  
  Then, the user client sends the REQUEST message again with the computed response value, username, nonce value and realm.
- **Step 4**: According to the username, the server extracts the user client’s password. Then the server verifies whether the nonce is correct. If it is correct, the server computes as Eq. (1) and checks whether the computed value is equal to the received response value. If they match, the server authenticates the identity of the client.

2.3. Security issues for SIP

Several researches have highlighted various security flaws and vulnerability caused by open network and the standard of SIP message [4–7]. Under this situation, SIP communications are susceptible to several types of attacks and threats. In this section, we summarize the security flaws and vulnerability with the following interpretation.

2.3.1. The standard of SIP message

It is well know that the standard of the SIP message using a text-based representation called Session Description Protocol (SDP) in RFC2327, which is similar to a HTTP message, and it can be either a request or acknowledge to a corresponding request, consisting of the header fields and optionally of a message body. The header can reveal information about the communication patterns and content of individuals or other confidential information. The SIP message body may also contain user information (media type, code, address and ports, etc.). Under this situation, SIP communications are susceptible to several types of attacks. The adversary can launch eavesdropping attack which is permitted to gain information on user identities, services types, media, and so on. This information can be used to perform other types of attacks. Modification attacks may be occurred under the assumption that the adversary intercepts the SIP message in order to change some service characteristics. This kind of attack can be used to hijack the signaling flow forcing a particular route, or to change a user registration or modify a service profile. In other words, the standard of SIP message lacks for securing all headers and parameters which would possible need protection in SIP message.

2.3.2. Signaling attacks

According to SIP signaling flow shown in Fig. 2, consider a case where the adversary captures the SIP traffic for a specific session. Such an eavesdropping attack may be obtained some information (e.g. identities of communication parties) by the adversary. Under this situation, the adversary can use the appropriate session parameters to create a spoofed BYE message, which aim to cause denial of service (DoS). These kinds of attacks are known as signaling attacks [4,12].

2.3.3. The flaws for HTTP Digest authentication protocol

As noted in RFC 2617, the current authentication mechanism in SIP, HTTP Digest based authentication mentioned previously, is limited in real application. We summarize the flaws as follows.

- **SI 1**: It easily suffers from server spoofing attack and Man-in-the-middle attack due to the lack of mutual authentication between the client and the server.
- **SI 2**: It uses the password as the authenticator. However, the password can be retrieved by way of starting the off-line password guessing attack with the incepted response value. The result will lead to the forgery attack.
- **SI 3**: It must previously configure the password table corresponding to the end users. The result will cause many problems, such as insider attack and stolen verifier attacks. Additionally, the pre-configured password table cannot apply to the end users with different network domains.
- **SI 4**: It cannot prevent from signaling attack. Since the important parameters in the header filed of SIP message (e.g. the tag in the FORM header and the tag in the TO header) is not protected, the adversary easily uses them to generate a spoofed SIP message (e.g. BYE or CANCEL, etc.).
- **SI 5**: It cannot provide media protection mechanism during call session. That is to say, the sensitive information can be intercepted and utilized.
3. Preliminaries

In this section, we briefly review the basic concepts on SCPKs and some related mathematical problems.

3.1. Elliptic curve cryptography (ECC)

Let \((G, +)\) denotes an additive cyclic group of prime order \(q\) generated by \(P\). In general, the group \(G_1\) is a subgroup of the additive group of points of an elliptic curve over finite field \(F_p\) together with the extra point \(O\) living “at infinity”. We simple define a non-super singular Elliptic curve \(E\) over \(F_p\) to be an equation of form
\[
y^2 = (x^3 + ax + b) \mod p,
\]
with \(a, b \in F_p\), satisfying \((4a^3 + 27b) \mod p \neq 0\).

and then we look at the points on \(E\) with coordinates in \(F_p\), which we denote by
\[
E(F_p) = \{(x, y) : x, y \in F_p \text{ satisfy } y^2 = x^3 + ax + b \} \cup \{O\}.
\]

The curve points on \(E(F_p)\) must obey the elliptic curve addition algorithm. In view of shortness, we omit the details and refer to \([21–23]\).

(A) Elliptic curve discrete logarithm problem (ECDLP)

Consider the equation \(Q = k \times P\), where \(Q \in G_1\) and \(k \in Z_q\). It is relatively easy to calculate \(Q\) given \(k\) and \(P\), but it is relatively hard to determine \(k\) given \(Q\) and \(P\).

(B) Elliptic curve Diffie–Hellman key exchange protocol (ECDH)

A key exchange between Alice and Bob can be accomplished as follows:

1. Alice generates a random number \(a \in Z_q\), calculates \(D_\text{A} = a \times P\) and sends \(D_\text{A}\) to B.
2. Bob generates a random number \(b \in Z_q\), calculates \(D_\text{B} = b \times P\) to A.
3. Alice can calculate key \(sk_\text{A} = a \times D_\text{B}\) and Bob can calculate key \(sk_\text{B} = b \times D_\text{A}\).

Since, \(a \times D_\text{A} = a \times b \times P = b \times a \times P = b \times D_\text{A}\), thus \(sk_\text{A} = sk_\text{B}\). To break this scheme, an attacker would face ECDLP, which is assumed hard.

3.2. Self-certified public keys (SCPKs)

A self-certified public key (SCPKs) is an efficient alternative to certificate-based PKI \([24,25]\). Instead of verifying public key using an explicit signature on a user’s public key, the public key is computed directly from the signature of the third trust party (TTP) on the user’s identity. The scheme is presented below.

1. **Setup:** It is assumed that all entities have globally agreed upon a non-singular high elliptic curve \(E\) defined over a finite field \(p\) (i.e., \(E(F_p)\)), which is used with a based point generator \(P\) of prime order \(q\). TTP chooses a key pair \((s_T, PK_T)\), where \(PK_T = s_T \times P\). We assume that the related parameters and \(PK_T\) are publicly and authenticably available.
2. **Private key generation:** To generate a key pair on Alice’s identity \(ID_\text{A}\), Alice chooses a random number \(k_\text{A} \in Z_q\) and computes \(K_\text{A} = k_\text{A} \times P\), and then sends \(K_\text{A}\) and \(ID_\text{A}\) to TTP over secure channel. After receiving \(K_\text{A}\) and \(ID_\text{A}\), TTP assigns a random number \(r_\text{A}\) and computes \(K_\text{A} = K_\text{A} + r_\text{A} \times P\), and then computes the signature parameter \(s_\text{A}\) as follows:
\[
s_\text{A} = h(ID_\text{A}, filed ID_\text{A}) \cdot s_T + r_\text{A}.
\]
Then, both \(s_\text{A}\) and \(R_\text{A}\) are transmitted to Alice who obtains her secret key as follows:
\[
s_\text{A} = s_\text{A} + k_\text{A}.
\]

3. (3) **Public key extraction:** Through the above pre-deployment, the corresponding public key \(PK_\text{A}\) can be computed by everyone who receives the public key parameter \(R_\text{A}\). Thus, \(PK_\text{A}\) can be obtained as following equation:
\[
PK_\text{A} = h(ID_\text{A}, filed ID_\text{A}) \times PK_T + R_\text{A}.
\]

Eq. (3) can be deduced as follows:
\[
PK_\text{A} = s_\text{A} \times P = (s_\text{A} + k_\text{A}) \times P = s_\text{A} \times P + k_\text{A} \times P = (h(ID_\text{A}, filed ID_\text{A}) \times s_T \times P + (r_\text{A} \times P + k_\text{A} \times P)
\]
\[
= (h(ID_\text{A}, filed ID_\text{A}) \times PK_T + R_\text{A}.
\]

4. Proposed authentication scheme

In this section, we propose a new secure password authenticated key agreement scheme using SCPKs on elliptic curve. The proposed scheme retains the original SIP authentication structure without the need of any password table for verification. In the meantime, our proposed scheme achieves mutual authentication for communication parties with different SIP domains. Moreover, we provide the password change phase to make the eligible user change password quickly and securely. The notations used through this paper are summarized in Table 1. Without loss of generality, the scheme consists of four phase: the setup phase, the registration phase, the mutual authentication phase and the password change phase. Different phases are stated as follows.

4.1. The setup phase

According to the setup requirements of SCPKs mentioned above, we define a trust authority (TA) in each SIP domain to issue the long-term private keys to the entities in the same domain. Suppose \((G, +)\) be an additive cyclic group of prime order \(q\) generated by \(P\). TA selects a random secret number \(s_T \in Z_q\) as his own long-term private key, and then computes the point multiplication \(s_T \times P\) as public key \(PK_T\). Then TA publishes the system parameters \((G, q, P, PK_T, h, H_1, H_2)\) and keeps \(s_T\) secret.

Next, each eligible server \(S_i\) chooses a random number \(k_i\), compute \(K_i = k_i \times P\), and then sends \((ID_i, K_i)\) to TA over a secure channel. Upon receiving \((ID_i, K_i)\), TA selects a random number \(t_i\) and computes the signature parameters \((R_i, s_i)\), in which

| Table 1 |
|---|---|
| Notation | Definitions |
| \(U_i\) | The \(i\)th user client |
| \(S_i\) | The \(i\)th server |
| UID_i | Unique SIP identity of \(U_i\) |
| SID_i | Unique SIP identity of \(S_i\) |
| UPW_i | A password chosen by the user client \(U_i\) |
| \(G_1\) | An additive cyclic group of a prime order \(q\) |
| \(P\) | Generator of group \(G_1\) |
| TA | The trust authority |
| \((s_T, PK_T)\) | The key pair of TA, where \(PK_T = s_T \times P\) |
| Q^* | The x-coordinate of elliptic curve point Q |
| \(h/\) | The secure one-way hash function \([0, 1]^n \rightarrow [0, 1]^m\), where \(n\) is the length of output |
| \(H_1(.)/H_2(.)\) | The suitable key derivation functions |
| \(\times\) | The scalar multiplication of elliptic curve point over \(E(F_p)\) |
| \(\|\) | The concatenation operation |
4.2. The registration phase

Before the new user client \(U_i\) becomes a member of SIP domain, she/he performs the following process with TA. The registration phase is shown in Fig. 4.

R1. The user client \(U_i\) chooses his identity \(UID_i\) and password \(UPW_i\). In addition, the user client \(U_i\) selects a random value \(k_i\) and computes \(K_i = k_i \times P\). After that, the user client submits \((UID_i, UPW_i, K_i)\) to TA over a secure channel.

R2. After receiving \((UID_i, UPW_i, K_i)\), TA generates a random number \(r_i \in Z_q^*\) and computes \(R_i = r_i \times P + K_i\). Then, TA computes the following parameters:

\[
\begin{align*}
\overline{s}_i &= h(UID_i || K^*_i) \cdot s_T + r_i, \\
m_i &= \overline{s}_i + h(UPW_i || K^*_i), \\
PK_i &= h(UID_i || K^*_i) \times PK_T + R_i.
\end{align*}
\]

After that, TA personalizes a smart card with the secret parameters \((m_i, R_i, K_i, PK_i, h(\cdot))\) and issues the smart card to the user client \(U_i\) over a secure channel.

R3. When the user client \(U_i\) receives the smart card, she/he inputs \((UID_i, UPW_i, k_i)\) to update \(m_i\) with the following steps: First, the smart card like-device computes \(s_{\overline{i}} = m_i - h(UPW_i || K^*_i)\) and \(s_i = s_{\overline{i}} + k_i\). After that, the smart card checks whether \(s_i \times P\) is equal to \(PK_i\). If they are equal, the smart card computes the new value \(m_i^{new} = m_i + k_i\) to update \(m_i\); otherwise, terminate the process for updating \(m_i\).

4.3. The mutual authentication phase

Once the user client \(U_i\) wants to communicate with the server \(S_j\), the mutual authentication phase is invoked and shown in Fig. 5. The smart card and the server \(S_j\) cooperate to complete this phase.

4.3.1. The login phase

First, the user client \(U_i\) inputs his identity \(UID_i\) and password \(UPW_i\), the smart card performs the following operations.

L1: Compute \(s_i' = m_i - h(UPW_i || K^*_i)\). Check whether \((s_i' \times P)\) is equal to \(PK_i\) stored in the smart card. If they are not equal, the login requests fails; otherwise, the smart card proceeds to the next step.

L2. Choose a random number \(a \in Z_q^*\) and compute \(T_i = a \times P\). After that, send the login request \((UID_i, R_i, T_i)\) to the server \(S_j\) over a public channel.

4.3.2. The server authentication phase

When receiving the login request \((UID_i, R_i, T_i)\) from the user client \(U_i\), the server \(S_j\) performs the following operations to challenge the user client \(U_i\).

SA1. Generate a random number \(b\) and computes \(T_j = b \times P\).

SA2. Compute the public key \(PK_j\) of the user client \(U_i\) using the received \(UID_i\) and \(R_i\) in accordance to Eq. (3). Then obtain the shared secret key \(x_j\) with the following equation:

\[
\begin{align*}
X_j &= b \times PK_j + (s_j + b) \times T_i, \\
\overline{x}_j &= H_1(X_j^i).
\end{align*}
\]

where \(H_1\) is a suitable key derivation function. Practical instantiations of \(H_1\) include \(H_1(z) = SHA - 1(01z)\).

SA3. Compute the message authentication code \(Auth_j\) with the following equations:

\[
\begin{align*}
X_j &= b \times PK_j + (s_j + b) \times T_i, \\
\overline{x}_j &= H_1(X_j^i).
\end{align*}
\]

SA4. Compute the public key \(PK_j\) of the server \(S_j, X_j\) and \(x_j\) with the following equations:

\[
PK_j = h(UID_i || K^*_i) \times PK_T + R_j, \\
X_j &= a \times PK_j + (s_i + a) \times T_j \quad \text{and} \quad x_j = H_1(X_j^i).
\]

SA5. Check the validity of the received message authentication code \(Auth_j\) with the computed value \(h(\text{nonce} || \text{realm} || T_j || T_j^* || UID_i || x_j)\) as response. After that, send \(\text{RESPONSE message (nonce, realm, Authj)}\) to the server \(S_j\).

4.3.3. The user authentication phase

After the server \(S_j\) is authenticated, the user client \(U_i\) sends the response message to the server \(S_j\) with the following step.

UA1. Compute the response value \(Auth_i = h(\text{nonce} + 1 || \text{realm} || UID_i || T_i^* || T_j^* || x_j)\) as response. After that, send \(\text{RESPONSE message (nonce, realm, Authi)}\) to the server \(S_j\).

---

Fig. 4. The registration phase of the user client.

Fig. 5. The mutual authentication phase of the user client.

Fig. 6. The login phase of the user client.

Fig. 7. The server authentication phase of the user client.

Fig. 8. The user authentication phase of the user client.
5. Correctness and security analysis

In this section, we will show the correctness of our scheme. Furthermore, the security analysis is examined.

5.1. Correctness

After the user client $U_i$ sends the login request message $(\text{UID}_i, R_i, T_i)$, only the server $S_i$ obtains the secret key $X_{ij} = b \times PK_i + (s_i + a) \times T_i$ and $x_j = H_1(X_{ij})$ as Eqs. (5) and (6). Similarly, after the server $S_i$ sends the CHALLENGE message (nonce, realm, $\text{SID}_i$, $R_i$, $T_i$, $\text{Auth}_i$), only the user client $U_i$ obtains the secret key $X_{ij} = a \times PK_i + (s_i + a) \times T_i$ and $x_j = H_1(X_{ij})$. The equality of $X_{ij}$ and $x_j$ and be proven as follows:

\[ X_{ij} = (a \times PK_i + (s_i + a) \times T_i) = (a \times s_i + b \times P + s_i \times b \times P + a \times b \times P) = (a \times s_i \times b \times P + a \times s_i \times b \times P + a \times b \times P) = (b \times PK_i + (s_i + b) \times a \times P) = (b \times PK_i + (s_i + b) \times T_i) = X_{ij} \]

The user client $U_i$ checks the validity of the received $\text{Auth}_i$ with $h(\text{nonce}||\text{realm}||\text{SID}||\text{UID}||x_j)$. If it holds, the user client $U_i$ believes no one besides the server $S_i$ to own the shared secret key $X_{ij}$. Thus, the legality of the server $S_i$ can be assured. Similarly, the server $S_i$ can check whether the response value $\text{Auth}_i$ is equal to $h(\text{nonce} + 1||\text{realm}||\text{SID}||\text{UID}||x_j)$. If yes, the user client $U_i$ is authenticated.

5.2. Salient features

According to the proposed scheme, the following features are summarized.

SF1. Our scheme uses the shared secret key $X_{ij}(X_{ij})$ to achieve the mutual authentication between the communication entities, where $X_{ij}(X_{ij})$ is composed of the long-term private key $s_i$ and the ephemeral private keys $a(b)$. No one besides the communication entities can obtain the shared secret key $X_{ij}(X_{ij})$.

SF2. Our scheme cannot let the corrupted TA impersonate other’s parties to gain benefit. If TA wants to obtain the long-term private keys $s_i$ to impersonate other user client $U_i$, he cannot work. According to SCPKs, TA cannot compute the long-term private key $s_i = s_i + k_i$ since the random value $k_i$ is selected by the user client $U_i$. On the other hand, TA cannot extract $k_i$ from the received $K_i$ on the security of ECDLP.
SF3. Our scheme provides a password-based authentication scheme without maintaining any password or verifier table in the server.

SF4. Our scheme provides a user-friendly password change option to the user client without any assistance from TA. This feature greatly reduces the network traffic and overhead on TA. On the other hand, once the smart card is lost, the adversary cannot enter new password without validating the original password.

5.3. Security analysis

We will show that our protocol not only provides complete authentication scheme for SIP-based service but also prevents the following possible attacks.

5.3.1. Preventing the replay attack

RA1. The replay attack means that the attacker replays the same message of the receiver or the sender again. The server $S_i$ sends a CHALLENGE message $(\text{nonce}. \text{realm}. \text{SID}_i, \text{nonce}_i)$ to the user client $U_i$, where $\text{nonce}$ is assumed to be generated independently and different in each session. If the attacker replays a previous CHALLENGE message $(\text{nonce}_i'. \text{realm}. \text{SID}_i, \text{nonce}_i')$ to impersonate the server $S_i$, the user client $U_i$ checks whether $\text{nonce}$ has been reused. Thus, the replay attack can be detected and rejected.

RA2. If the attacker replays a previous login request message $(\text{UID}_i, \text{realm}. \text{Auth}_i)$ and corresponding RESPONSE message $(\text{nonce}. \text{realm}. \text{Auth}_i')$ to impersonate the user client $U_i$, the server $S_i$ believes the attacker to be illegal, since $\text{Auth}_i'$ is not equal to the computed value $h(\text{nonce} + 1 | \text{realm}| \text{UID}_i | \text{SID}_i | \text{T}_i | \text{x}_i)$. Thus, our scheme can withstand the replay attack.

5.3.2. Preventing the forgery attack

When the adversary wants to masquerade as the user client $U_i$ to pass the verification of the server, he/she cannot construct a valid message authentication code $\text{Auth}_i = h(\text{nonce} + 1 | \text{realm}| \text{UID}_i | \text{SID}_i | \text{T}_i | \text{x}_i)$ without knowing the ephemeral secret key $X_p$, since $X_p$ only can be derived by the user client $U_i$. Similarly, the adversary cannot masquerade as the server $S_i$ without the knowledge of the ephemeral secret key $X_p$.

5.3.3. Preventing the offline password guessing attack

The adversary may start the offline guessing attack to obtain the user's password. In our scheme, the password is stored in the smart card and used to extract the long-term secret key of the user. If the adversary wants to intercept the transmitting SIP message to guess the corresponding password, it cannot work since the password is not contained in the SIP message.

5.3.4. Preventing the man-in-the-middle attack

In our scheme, the communication entities can achieve mutual authentication under the assumption of sharing the ephemeral secret key $X_p$ between them. Thus, the adversary cannot launch the man-in-the-middle attack to cheat either the user client or the server.

5.3.5. Preventing the insider attack

There exists an inherent risk of secret information being stolen. We discuss the secret information leaked with the following two situations.

IA1. There exists an inherent risk of passwords being stolen. In our scheme, the server $S_i$ does not require to store the user password or verifier table, thus eliminating such risks.

IA2. There exists an inherent risk of the long-term private key being leaked. In our scheme, the long-term private keys are decided by the user client. Even though the private key $s_i$ of TA is leaked, the insider with privilege still cannot compute the long-term private keys of other entities. In other word, the insider attack proves to be useless.

5.3.6. Preventing signaling attack

As mentioned in Section 2.3, the adversary can intercept the important parameters of SIP message (e.g., identities of communication parties) during the process of call setup. Upon that, the adversary generates the corresponding message (e.g., SIP BYTE or CANCEL) and sends it to the appropriate SIP server in order to cause an illegal termination-alteration of the session. The result will directly and substantially affect VoIP service availability and reliability. To achieve the integrity and authenticity, our scheme uses Hashing for Message Authentication code (HMAC) with the shared secret keys concerned with the long-term private key of communication entities. Hence, the adversary cannot create a valid SIP message to masquerade as the user agents without knowing their long-term private keys.

5.3.7. Smart card security

In several literatures [26,27], they discussed the security about smart cards. They assumed that the secrets stored in a smart card may be breached, so that they presented some weaknesses or attacks. In this paper, we do not focus on the security about smart cards. We use smart card to aid the users to memorize their long-term private keys. Certainly, one self-protected mechanism is should be provided to securely store these message on the smart card [28]. In additions, once a user loses his smart card, he/she should report it to his corresponding TA.

5.3.8. Session key security

After the communication entities achieve mutual communication, the session key can protect the later sensitive information between them. Next, we show the security strength against possible session key attacks.

(A) Known-key security: The known-key security is defined as the assurance that any future session keys will not compromised if the current session key will be known to an attacker. In our scheme, the user client $U_i$ and the server $S_i$ generate a unique session key $SK_p$, which is based on ECDH. Thus, the session key generated in each session is independent and should not be exposed if other session keys are compromised.

(B) Perfect forward secrecy: The perfect forward secrecy is defined as the assurance that any previous session keys will not compromised even though the secret information of the system is leaked. In our scheme, any session key between communication entities is related to nonce $a$ and $b$, which are unconfienced with the secret information, such as the long-term private keys of the communication entities and TA. Thus, our scheme can achieve perfect forward secrecy. In other words, even though TA's long-term private key (i.e., $s_i$) is compromised, this should not compromise the previously established session keys. This is known as TA forward secrecy and the key escrow problem can be avoided.

5.3.9. Data integrity

A system supporting data integrity implies that it can check if the data received from the client are correct. To prevent the transmitting SIP message from being modified, it is necessary that the
mechanism of data integrity can be introduced. In our scheme, the shared secret key $X_0$ is obtained during the mutual authentication process. If the related fields contained the transmitting SIP message need to be protected, HMAC involving the shared secret key $X_0$ can be used instead of the complex public key mechanism.

6. Performance considerations and functionality comparison

In this section, we will evaluate the performance of our proposed scheme. In general, the performance evolution usually is divided into communication cost and computation cost. As we all know, an ECC with 160-bit key length could offer roughly the same level of security as RSA with 1024-bit modulus. We divide the computation cost of our scheme into two parts: offline computation and online computation, where the offline computation cost can reduce the latency between the communication parties. In mutual authentication phase, we subdivide it into four parts, including the login phase, the server authentication phase, the user authentication phase and session key agreement phase. In login phase, our scheme requires one hashing operation and two scalar multiplications of elliptic curve. In the server authentication phase, our scheme needs six hashing operation, seven scalar multiplications of elliptic curve, and four point additions of elliptic curve. In the user authentication phase, our scheme needs two hashing operations. In session key agreement phase, our scheme needs two hashing operations and two scalar multiplications of elliptic curve. In view of efficiency computation, our scheme is more efficient than other schemes in [9–11], since our scheme does not involve costly IBS signature, bilinear pairings and modular exponentiation. Some previous implementations of elliptic curve cryptographic primitives on smart cards or microprocessors have been developed [29–31]. Recently, Scott et al. actually evaluate the cost of one scalar multiplication with the Philips HiPersmart card, where the processor of HiPersmart card offers a maximum clock of 36 MHz and 16K RAM memory [32]. In which, $G_1$ is a subgroup of order $q$ on an elliptic curve over a finite field $\mathbb{F}_p$, where $p$ is a 512-bit prime and $q$ is a 160 bit prime. Under this situation, the time spent in scalar multiplication of elliptic curve (i.e., $T_{\text{mec}}$) is around 270 ms. We ignore some light-weight operations including modular addition in $Z_q$ and simple hashing operation. On the other hand, to simplify the estimation of the communication cost, our scheme is compared with HTTP Digest authentication scheme for SIP. According to the basics of ECC, the points on $E(F_p)$ are made up of $(x, y)$ coordinates and the y-coordinate can be compressed with a compress bit. Therefore, it suffices to stores $x$ and a compression bit $b$. This point compression is accomplished using the function $\text{COMPRESS}$. That is to say, the points on $E(F_p)$ can be represented with 513 bit-length.

In our scheme, the communication cost includes the capacity of transmitting message between the user client and the server. In contrast to HTTP Digest authentication scheme for SIP, our scheme does not change the structure of SIP message except for adding some fields, including message hash code, the public key parameter and the ephemeral public key. The user client $U_i$ sends both the message $(UID, R_i, T_i)$ and $(nonce, realm, Auth_i)$ to the server $S_j$. The extra capacity of corresponding messages $(R_i, T_i, Auth_i)$ is 1186 bits (i.e., $513 + 513 + 160$). Similarly, the server sends $(nonce, realm, SID, R_i, T_i, Auth_i)$ to the user client. The extra capacity of corresponding messages $(R_i, T_i)$ is 1026 bits (i.e., $513 + 513$). Here, we summarize the performance evaluation of our scheme in Table 2. In addition, the functionality comparison between our scheme and other related schemes is listed in Table 3. It is obvious that our scheme can not only solve the security flaws of SIP-based service and is applied to authenticate the user agents with limited computing capability.

<table>
<thead>
<tr>
<th>Table 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>The performance evolution of our scheme.</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>User client side</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Online</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Offline</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Server side</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Online</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Offline</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>The time spent in the login phase</td>
</tr>
<tr>
<td>$2T_b + T_{\text{mec}} + T_{\text{arc}}$</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>$3T_b + 3T_{\text{mec}} + T_{\text{arc}}$</td>
</tr>
<tr>
<td>$T_b$</td>
</tr>
<tr>
<td>$0$</td>
</tr>
<tr>
<td>$3T_b + 3T_{\text{mec}} + T_{\text{arc}}$</td>
</tr>
<tr>
<td>$T_b$</td>
</tr>
<tr>
<td>$0$</td>
</tr>
<tr>
<td>$T_b + T_{\text{mec}}$</td>
</tr>
<tr>
<td>$5T_b + 5T_{\text{mec}} + T_{\text{arc}} \approx 1.35 \text{ s}$</td>
</tr>
<tr>
<td>$T_{\text{mec}} + T_{\text{arc}} \approx 0.27 \text{ s}$</td>
</tr>
<tr>
<td>Extra communication cost in contrast to HTTP Digest authentication (bits)</td>
</tr>
<tr>
<td>$1186 \approx 1.16K$</td>
</tr>
<tr>
<td>$1026 \approx 1K$</td>
</tr>
</tbody>
</table>
Acknowledgements

The authors thank the reviewers for their valuable comments and suggestions.

References