A New Provably Secure Authentication and Key Agreement Mechanism for SIP Using Certificateless Public-key Cryptography[1]

Fengjiao Wang[1, *], Yuqing Zhang[2]

[1,2]National Computer Network Intrusion Protection Center， GSCAS，
Beijing， 100049

Abstract. The session initiation protocol (SIP) is considered as the dominant signaling protocol for calls over the internet. However, SIP authentication typically uses HTTP digest authentication, which is vulnerable to many forms of known attacks. This paper proposes a new secure authentication and key agreement mechanism based on certificateless public-key cryptography, named as SAKA, between two previously unknown parties, which provides stronger security assurances for SIP authentication and media stream, and is provably secure in the CK security model. Due to using certificateless public key cryptography, SAKA effectively avoids the requirement of a large Public Key Infrastructure and conquers the key escrow problem in previous schemes.

Key words: SIP, certificateless public-key cryptography, authentication, key agreement

1 Introduction

SIP [1] is a signaling protocol based on the application-layer for establishing, modifying and terminating multimedia user sessions, and it is capable of operating on TCP or UDP and handles all the signaling requirements of a VoIP session. SIP messages are text-based and similar to HTTP format. The task of SIP is to establish streaming connection between hosts.

With the widespread use of VoIP in worldwide, SIP is currently receiving much attention. It seems to be the most promising candidate for call setup signaling for future IP-based telephony services, and it has been chosen by the Third-Generation Partnership Project (3GPP) as the protocol for multimedia application in 3G mobile networks. As SIP is being used more, the security of it is an important and urgent

[1] This work is supported by National Natural Science Foundation of China (grant Nos. 60573048, 60373040).

[*]Corresponding author .Tel:+86-010-68860988, Fax: +86-010-68860988.
E-mail address: wangfj@nipc.org.cn.

issue to ensure that every SIP-based Internet services can meet the corresponding requirements.

This paper concentrates on the security flaws in current SIP authentication procedure. We propose a new secure authentication and key agreement mechanism using certificateless public key cryptography [2], named as SAKA, which provides stronger security assurances for SIP.

1.1 Related Work

As noted in RFC2617 [3], the current authentication mechanism in SIP, HTTP digest based authentication, is vulnerable to many forms of attacks. In [4], Salsano et al pointed out that the HTTP digest authentication in SIP suffers from two major weaknesses when it is applied in SIP. One is the lack of securing all headers and parameters in SIP which would possibly need protection. The other is the requirement of pre-existing user configuration on servers. Furthermore, a methodology for the evaluation of the processing cost of SIP authentication procedure is also given in their work, which is of great meaning. Since the current authentication mechanism is not providing security at an acceptable level, several new schemes are proposed to improve it. The off-line password guessing attack and server spoofing attack to original authentication mechanism have been found in [5], and a new authentication scheme has been given to solve these problems, which is also immune to replay attack. In [6], an authentication scheme in SIP is developed by Srinivasan et al. Their proposition assumes that proxy server authenticates user client with registrar server, which leads to a requirement that proxy server and registrar server are trusted. Furthermore, a lightweight scheme for SIP user authentication and securing the integrity of SIP contact addresses is proposed in [7], which proposes that user client phones do the signing of their contact addresses instead of the registrar server. At meantime, this scheme assumes that the registrar servers have pre-issued certificates issued by trusted authority, and that the SIP servers in both calling party and called party domain trust each other. The concrete advantages and disadvantages of [6, 7] are analyzed in [8].

Different from the above mentioned work, recently, a new authentication mechanism and key agreement protocol for SIP using Identity-based cryptography has been given in [9], which provides mutual authentication and provably secure key agreement protocol between previously unknown parties, and avoids an expensive PKI due to the usage of Identity-based cryptography. While this new scheme entails a trusted authority (TA) in each security domain to issue private keys, thus a key escrow facility is also needed for law enforcement and makes it only available in a security domain environment.

1.2 Our Contribution

Motivated by [9], we propose a new secure authentication and key agreement mechanism based on certificateless public key cryptography, which achieves mutual authentication and key agreement in SIP, and is provably secure in CK security model

[10]. Simultaneously, our scheme conquers the key escrow and peer-to-peer communication problems in [9].

1.3 Outline of the Paper

The paper is organized as follows. Section 2 presents some background information on current authentication procedure in SIP. In section 3, certificateless public key cryptography is briefly introduced. In section 4, SAKA is presented with the security proof in CK security model, and the security attributes as well as the immunity to main attacks are analyzed. The advantages and limitations are discussed by comparing with the current solutions in section 5. Finally, we conclude the paper.

2 SIP Authentication Procedure

SIP authentication security is based on the challenge-response model, REGESTER and INVITE are two most commonly used SIP exchanges to connect to the network and establish a call respectively.

SIP authentication scheme works similarly to HTTP Digest authentication, in which a nonce value is used in challenging the target. The response includes then a checksum of the username，password，nonce value, HTTP method and requested URI, which provides protection from replay attacks. The concrete procedure is shown as Fig.1. Furthermore, SIP has two authentication dialogs: 401-Unauthorized and 407-Proxy Authentication Required. 401 responses are mainly used during REGISTER, while 407 responses are used during call establishment with intermediary SIP proxies (predominately during INVITE).

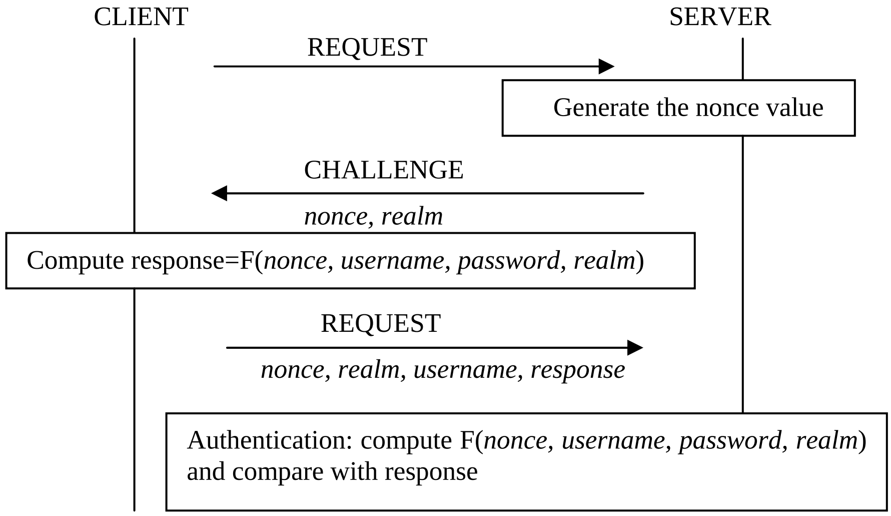
CLIENT
SERVER
REQUEST
Generate the nonce value
CHALLENGE
nonce, realm
Compute response=F(nonce, username, password, realm)
REQUEST
nonce, realm, username, response
Authentication: compute F(nonce, username, password, realm) and compare with response

Fig.1. Digest authentication procedure in SIP.

However, this digest authentication doesn't meet the security requirements in SIP-based IP telephony service, and its security flaws urge us to improve it.

3 Certificateless Public Key Cryptography

The concept of certificateless public key cryptography (CL-PKC) was first proposed by Al-Riyami and Paterson in 2003 [2]. A CL-PKC system makes use of a trusted third party (TTP) which is named as the key generating centre (KGC) to supply an entity *A* with a partial private key D_A which the KGC computes from an identifier ID_A. The entity *A* then combines its partial private key D_A with some secret information to generate its actual private key S_A. In this way, *A*'s private key is not available to the KGC. Unlike id-based cryptography, the public key is no longer computable from an identity (or identifier) alone. Instead, *A* combines its secret information with the KGC's public parameters to compute its public key P_A, and P_A might be made available to other entities by transmitting it along with messages or by placing it in a public directory. But no further security is applied to the protection of *A*'s public key. In particular, there is no certificate for *A*'s key. The structure of CL-PKC ensures that the key can be verified without a certificate. To encrypt a message to *A* or verify a signature from *A*, one must know P_A and ID_A.

In contrast to traditional public key cryptographic systems, CL-PKC does not require the use of certificates to guarantee the authenticity of public keys. Similar to ID-PKC, CL-PKC does rely on a trusted third party (TTP) to generate a master key and the corresponding problems related to bilinear maps. On the other hand, CL-PKC doesn't suffer from the key escrow property that seems to be inherent in ID-PKC. Thus CL-PKC can be seen as a model for the use of public key cryptography that is intermediate between traditional certificated PKC and ID-PKC. To know more about the CL-PKC, the reader is referred to [2].

4 SAKA

As stated in [9], a key agreement protocol is required to establish shared secret for SRTP [13] between previously unknown parties, however, no such facility exists. Thus, we propose a new authentication and key agreement mechanism, which achieves not only the authentication functionality but also a shared master key between previously unknown parties, and provides stronger security assurance. Parallel to the authentication mechanism based on ID-based cryptography in [9], SAKA uses certificateless public key cryptography.

4.1 System Initiation

Certificateless public key cryptography is based on bilinear maps, G_1 denotes an additive group of prime order q and G_2 a multiplicative group of the same order, and we let P denote a generator of G_1. To be concise, we omit the description of bilinear maps related knowledge in this paper, to know more please refer to [2].

According to the setup requirements of CL-PKC, we define a key generating centre (KGC) in each security domain of SIP to issue partial private keys for entities in the same domain. To provide stronger security, we adopt the binding technique [2] which ensures that users can only create one public key for which they know the corresponding private key in this paper.

Each entity A has a SIP identity ID_{A}. The concrete initiation process is depicted by the following five randomized algorithms.

(1). Setup: This algorithm first calls the BDH parameter generator IG with security parameter $\boldsymbol{k}$ to generate output $< G_{1,}\ G_{2,}\ \hat{e} >$, and choose an arbitrary generator $P \in G_1$; then returns the system parameters **params** and master-key s. Usually，this algorithm is run by the KGC. We assume that $< G_{1,}\ G_{2,}\ \hat{e} >$, **params** and P_0=sP are publicly and authentically available, but that only the KGC knows **master-key**.

(2). Set-Secret-Value: This algorithm takes as inputs **params** and an entity A's SIP identity ID_{A} as inputs and outputs A's secret value x_{A}.

(3). Set-Public-Key: This algorithm takes **params** and entity A's secret value x_{A} as input and from these constructs the public key P_{A} for entity A, $P_{\mathrm{A}} = <X_{\mathrm{A}},Y_{\mathrm{A}}>$， $X_{\mathrm{A}} = x_{\mathrm{A}}P$, $Y_{\mathrm{A}}= x_{\mathrm{A}}sP$.

(4). Partial-Private-Key-Extract: This algorithm takes **params**, **master-key** and entity A's SIP identity ID_{A}, $ID_{\mathrm{A}} \in \{0,1\}^*$, as input. It returns a partial private key D_{A}, $D_{\mathrm{A}} = sQ_{\mathrm{A}}$, where Q_{A} is defined to be $Q_{\mathrm{A}}=H_1(ID_{\mathrm{A}}||P_{\mathrm{A}})$.Usually this algorithm is run by the KGC and its output D_{A} is transported to entity A over a confidential and authentic channel.

(5). Set-Private-Key: This algorithm takes **params**, an entity A's partial private key D_{A} and A's secret value x_{A} as input. The value x_{A} is used to transform D_{A} into the (full) private key S_{A}. The algorithm returns S_{A}, $S_{\mathrm{A}} = x_{\mathrm{A}}D_{\mathrm{A}}$.

Each entity in a security domain, including the server, sets its public and private keys according to the algorithms introduced above, if it is necessary. In addition, we define a hash function H, which is publicly available.

4.2 A New Key Agreement Protocol for SIP

The key agreement protocol for SIP [9] using ID-based cryptography was provably secure in BR93 [11], whose adversary is restricted to be benign. That is, if an attacker wishes the attacked party to output a ”acceptance” decision, then his behaviors are restricted to be benign[12], who passes the messages honestly between the oracles at his choice. Additionally, the key agreement protocol in [9] is vulnerable to the colluding attack, when the TA's cooperate to obtain the private key of the users. Therefore, we propose a new secure key agreement protocol using certificateless public key cryptography, which is provably secure in CK model and avoids the colluding attack.

4.2.1 A New Key Agreement Protocol

The initialization for SAKA is formally specified using the five algorithms mentioned above. Entities *A* and *B* who wish to agree a key (the two participants may be in various security domains), they first each choose random values a, $b \in_R \mathbb{Z}_q^*$. Given these initializations, the key agreement protocol: Protocol 1 is shown as follows:

Protocol1. Key agreement protocol

$$\begin{array}{ccc}
\textbf{\textit{A}} & & \textbf{\textit{B}} \\
a \in_R \mathbb{Z}_q^* & & b \in_R \mathbb{Z}_q^* \\
 & \xrightarrow{s, T_A = aP, P_A} & \\
 & \xleftarrow{s, T_B = bP, P_B, Sig_{S_B}(s, A, T_A, T_B, P_B)} & \\
 & \xrightarrow{s, Sig_{S_A}(s, B, T_B, T_A, P_A)} & \\
K_{AB} = \hat{e}(S_A, T_B) \cdot \hat{e}(Q_B, aY_B) & & K_{BA} = \hat{e}(S_B, T_B) \cdot \hat{e}(Q_A, aY_A) \\
SK_{AB} = H(K_{AB}) & & SK_{BA} = H(K_{BA}) \\
 & SK_{AB} = SK_{BA} &
\end{array}$$

When the above messages have been exchanged, both users check the validity of each other's public keys in the usual way and the signatures. Then, *A* computes $K_{AB} = \hat{e}(S_A, T_B) \cdot \hat{e}(Q_B, aY_B)$ and *B* computes $K_{BA} = \hat{e}(S_B, T_B) \cdot \hat{e}(Q_A, bY_A)$, where P_A =< X_A , Y_A >, s is a session identifier. It is easy to see that $K = K_{AB} = K_{BA} = \hat{e}(bs_A + as_B, P)$ is a key shared between *A* and *B*; their shared session key is then $SK_{AB}=H(K_{AB})$ where *H* is a suitable hash function.

4.2.2 Security Proof of the Key Agreement Protocol

We prove the security of protocol 1 above using the security model proposed by Canetti and Krawczyk in [10], which adopts the modular design and analysis of key exchange protocol, and thus simplifies the difficulty of design and analysis of a secure protocol.

Adversary model: U= {P_1, P_2,…, P_n}is the set of participants. From the adversary's point of view, each participant in the protocol is an Oracle. In the unauthenticated-links model (UM), the adversary is a (probabilistic) polynomial-time machine with full control of the communication between parties. The ability of the attacker is modeled by the queries to the oracles. It is assumed that an attacker μ repeats the following choice operations till the end of a protocol run in UM:

(1). Active P_i to execute Π : μ actives Π with an action request q or an incoming message m with a specified sender P_i. The effect is
   1) P_i runs its program and hands the outgoing messages and action requests to μ ;
   2) Local outputs are known to μ .

(2). Corrupt P_i: The effect is
   1) μ learns the current states of P_i ;
   2) 'P_i is corrupted'is added to P_i's local output;
   3) P_i is no longer activated.

(3). Issue a session-state reveal for a special session within some party P_i:
   1) μ learns the current states of the specified session within P_i;
   2) This event is recorded (in P_i's local output).

(4). Issue a session-output query for a special session within some party P_i:
   1) μ learns the ”secret” output of the specified session within P_i;
   2) This event is recorded (in P_i's local output).

Additionally, sessions can be expired in CK model. Once a session is expired the attacker is not allowed to perform a session-output query or a state-reveal against the session, but is allowed to corrupt the party that holds the session. Protocols are said to enjoy “perfect forward secrecy”, if the expired sessions are protected even if party corruption exists.

An adversary model called authenticated-links model (AM) is defined in the same way as the UM, but a difference exists: the attacker is restricted to only deliver messages truly generated by the parties without any change or addition to them. To capture the equivalence of functionality between protocols in different adversary models, the notion of “emulation” is introduced between the UM and AM particularly.

Security goals: the security definition in CK model is based on indistinguishability, the “success” of an adversary μ is measured via its ability to distinguish the real values of session keys from independent random values, and this ability is formalized by the notion of a test-session query. The test-session query of μ proceeds as follows:

(1). μ executes a series of reasonable operations, then he chooses a session arbitrarily (completed, unexpired, unexposed) whose session key is noted as *K;*

(2). $b \in_R \{0,1)$, if b=0, return K to μ , or else μ gets $K^{'}$ ($K^{'} \in_R S_K$);

(3). μ proceeds to execute other reasonable operations(except expose the test-session);

(4). μ outputs the guess to b.

Now we address the security of a protocol in CK model by the following definition：

Definition 1. (SK-secure) A key establishment protocol Π (in UM) is SK-secure, if for any UM adversary μ , Π satisfies the following two conditions:

(1). If P_i, P_j (uncorrupted)have completed a matching session, both of them output the same session key;

(2). $\Pr[guess_{\mu} = b] < 1/2 + \varepsilon(k)$, where $\varepsilon(k)$ is a negligible function.

Similarly, for any an adversary in AM the above conditions are satisfied, we say that Π is SK-secure in AM.

Security proof: We first prove the security of a key agreement protocol 2 in the authenticated-link model in CK, which adds a session identifier to each flow to the key agreement protocol in [9]. Then, we use an authenticator based on signature to transform it to our new protocol, which has the same security in unauthenticated-link model. Protocol 2 is shown as follows:

Protocol 2 Key agreement protocol

$$\begin{array}{ccc}
\boldsymbol{A} & & \boldsymbol{B} \\
a \in_R \mathbb{Z}_q^* & & b \in_R \mathbb{Z}_q^* \\
 & \xrightarrow{s, T_A = aP, P_A} & \\
K_{AB} = \hat{e}(S_A, T_B) \cdot \hat{e}(Q_B, aY_B) & \xleftarrow{s, T_B = bP, P_B} & K_{BA} = \hat{e}(S_B, T_B) \cdot \hat{e}(Q_A, aY_A) \\
SK_{AB} = H(K_{AB}) & & SK_{AB} = H(K_{AB}) \\
 & SK_{AB} = SK_{BA} &
\end{array}$$

Bilinear Diffie-Hellman(BDH) problem: Let P be a generator of G_1. The BDH problem in $< G_1, G_2, \hat{e} >$ is that given (P, xP, yP, zP) for some $x, y, z \in \mathbb{Z}_q^*$, compute $W = \hat{e}(P, P)^{xyz} \in G_2$.

Theorem1. Protocol 2 is SK-secure in the AM, assuming that BDH problem (for the pair of groups G_1, G_2) is hard and provided that H is a random oracle.

Proof: Condition 1 in Definition 1 follows from the assumption that the two oracles follow the protocol and adversary $\mathcal{A}$ is passive. In this case, both oracles accept (since they both receive correctly formatted messages from the other oracle) holding the same session key. In addition, the session identifier *s* uniquely binds the values of aP and bP to these particular matching sessions and differentiates them from the messages that the parties may exchange in other sessions.

As far as condition 2 is concerned, we assume that an AM KE-adversary $\mathcal{A}$ can guess the value of *b* correctly at the end of a test-session query with a non-negligible advantage ε. Assume that there exists an oracle $\Pi^n_{\mathcal{I},\mathcal{J}}$, which has a matching session to another oracle $\Pi^t_{\mathcal{J},\mathcal{I}}$, and holds the session key with the form $H(\hat{e}(jS_I + iS_J, P))$ for *i* chosen randomly by $\Pi^n_{\mathcal{I},\mathcal{J}}$ and *j* chosen at random by $\Pi^t_{\mathcal{J},\mathcal{I}}$. We say that $\mathcal{A}$ succeeds (against $\Pi^n_{\mathcal{I},\mathcal{J}}$) if at the end of $\mathcal{A}$'s experiment, $\mathcal{A}$ picks $\Pi^n_{\mathcal{I},\mathcal{J}}$ to ask a Test query and outputs the correct bit guess. Thus, by assumption for some non-negligible function η(k) ,

$$\Pr[\mathcal{A}\text{ succeeds}]=1/2+\eta(\text{k}) .$$

Define an event A_k that H has been queried on $\hat{e}(jS_I+iS_J,P)$ by $\mathcal{A}$ or some oracle other than $\Pi^{n}_{I,\mathcal{J}}$ or $\Pi^{t}_{\mathcal{J},I}$ .Then,

$$Pr[\mathcal{A}\text{ succeeds}]=Pr[\mathcal{A}\text{ succeeds}|A_k\ ]Pr[A_k\ ]+Pr[\mathcal{A}\text{ succeeds}|\bar{A}_k\ ]Pr[\bar{A}_k\ ] .$$

Since H is a random oracle, and $\Pi^{n}_{I,\mathcal{J}}$ and $\Pi^{t}_{\mathcal{J},I}$ remain fresh, $Pr[\mathcal{A}\text{ succeeds}|\bar{A}_k\ ]=1/2$. Thus,

$$1/2+\eta(k)\le Pr[\mathcal{A}\text{ succeeds}]=Pr[\mathcal{A}\text{ succeeds}|A_k\ ]Pr[A_k\ ]+1/2 ,$$

so that $Pr[A_k\ ]\ge\eta(k)$. Therefore, we say that if $\mathcal{A}$ succeeds in the test session when picking $\Pi^{n}_{I,\mathcal{J}}$ that has had a matching session to $\Pi^{t}_{\mathcal{J},I}$, then the probability that H has previously been queried on $\hat{e}(jS_I+iS_J,P)$ by $\mathcal{A}$ or some oracle other than $\Pi^{n}_{I,\mathcal{J}}$ or $\Pi^{t}_{\mathcal{J},I}$ is non-negligible.

Therefore, we construct an algorithm $\mathcal{D}$ which solves the BDH problem with non-negligible probability using $\mathcal{A}$ as a subroutine. The description of $\mathcal{D}$ is as follows:

Goal: on input the two groups G_1, G_2 , the bilinear map $\hat{e}$, a generator of G_1 , P and a triple of $P_0=xP, Q_A=yP, Q_B=zP\in G_1$ with $x,y,z\in\mathbb{Z}_q^*$, $\mathcal{D}$'s task is to compute and output the value $\hat{e}(P,P)^{xyz}$.

Operation: $\mathcal{D}$ Chooses $I,J\in_R U$ (the probability of picking a particular pair is $1/T_2(k)^2$), $n,\ t\in_R\{1,...,T_2(k)\}$ (the probability of picking a particular session is $1/T_2(k)^2$), and $\ell\in_R\{1,...,T_3(k)\}$ (the probability of choosing a particular value is $1/T_3(k)$), where $T_2(k)$ denotes polynomial bounds in the security parameter k on the number of sessions an oracle may enter into with another oracle, for some polynomial function T_2 , and $T_3(k)$ denotes polynomial bounds in the security parameter k on the number of distinct H queries made by $\mathcal{A}$ and its oracles for some polynomial function T_3 . $\mathcal{D}$ guesses that $\mathcal{A}$ will select $\Pi^{n}_{I,\mathcal{J}}$ to ask its Test query after $\Pi^{t}_{\mathcal{J},I}$ has had a matching session to $\Pi^{n}_{I,\mathcal{J}}$, and also guesses that the lth distinct H call made during the experiment will be on $\hat{e}(P,P)^{xyz}$. $\mathcal{D}$ simulates the running of setup algorithm (run by KGC) by choosing xP as $P_0=sP$, choosing all participants' secret values and computing the corresponding public and private keys, but with the exception of I and J's keys. e.g., for participant I, the public key is $<X_I=x_IP, Y_I=x_IsP>$ .As public values for I and J, $\mathcal{D}$ chooses yP as I's public key $X_I = x_I P$, and *zP* as *J*'s public key, $X_J = x_J P$. $\mathcal{D}$ then starts $\mathcal{A}$ and proceeds as follows:

(1). Invoking $\mathcal{A}$ on a simulated interaction in the AM with parties running Protocol 1. Hand $\mathcal{A}$ all the public available values *q*, < G_1, G_2, $\hat{e}$ >, *P* ,P_0=*sP*, and the public keys of each participants as the public parameters for the protocol execution;

(2). During the period of $\mathcal{A}$'s attacking experiment, $\mathcal{D}$ answers $\mathcal{A}$'s *Hash* queries at random, just like a real random oracle would; and answers ***Corrupt*** queries, ***Reveal*** queries and ***Send*** queries as specified by a normal oracle, except that if $\mathcal{A}$ asks *I* or *J* ***Corrupt*** queries and ***Reveal*** queries, $\mathcal{D}$ gives up; If $\mathcal{A}$ asks $\Pi^{n}_{I,J}$ or $\Pi^{t}_{J,I}$ ***Send*** queries, $\mathcal{D}$ answers $(1/2)Q^{-}_{I} zP$ and $(1/2)Q^{-}_{J} yP$ respectively.

(3). Whenever $\mathcal{A}$ activates a party to establish a new session (except for the ***l*th** session) or to receive a message, $\mathcal{D}$ follows the instructions of protocol 1 on behalf of that party. When a session is expired at a player, erase the corresponding session key from that player's memory. When a party is corrupted or a session (other than the ***l*th** session) is exposed, hand $\mathcal{A}$ all the information corresponding to that party or session as in a real interaction.

(4). When the ***l*th** session, say ($\Pi^{n}_{I,J}$, $\Pi^{t}_{J,I}$, *s*) is invoked with *I* to establish a key with *J*, let *I* send the message($s, T_I = iP, < X_I, Y_I >$) to *J*.

(5). When *J* is invoked to receive ($s, T_I = iP, < X_I, Y_I >$), let *J* send the message ($s, T_J = jP, < X_J, Y_J >$) to *I*.

(6). If the ***l*th** session ($\Pi^{n}_{I,J}$, $\Pi^{t}_{J,I}$, *s*)is ever exposed, or if $\mathcal{A}$ halts without choosing a test- session, or if $\mathcal{A}$ does not make its queries in such a way that $\Pi^{n}_{I,J}$ has a matching session to $\Pi^{t}_{J,I}$, or if $\mathcal{A}$ and its oracles do not make *l* distinct *H* oracle calls before $\mathcal{A}$ asks its Test query, then $\mathcal{D}$ gives up. Otherwise, $\Pi^{n}_{I,J}$ will accept (holding the key in a form of $H(\hat{e}(jx_I Q_I + ix_J Q_J, xP)) = H(\hat{e}((1/2)Q^{-}_{I} zPx_I Q_I + (1/2)Q^{-}_{J} yPx_J Q_J), xP)$ = $H(\hat{e}(P,P)^{xyz})$.

(7). If $\mathcal{A}$ halts and outputs a bit *b*, then $\mathcal{D}$ halts and outputs the ***l*th** distinct hash call as its guess at $\hat{e}(P,P)^{xyz}$.

Therefore, if the AM adversary $\mathcal{A}$ can guess the value of *b* correctly at the end of a test-session query with a non-negligible advantage, then $\mathcal{D}$ can guess $\hat{e}(P,P)^{xyz}$ with a probability $Pr[A_k] / T_1(k)^2 T_2(k)^2 T_3(k) \geq \eta(k) / T_1(k)^2 T_2(k)^2 T_3(k)$, which is non-negligible, and this contradicts the BDH assumption. This completes the proof of Theorem 1.

A signature-based MT-authenticator is shown as follows:

Signature-based MT-authenticator:
1. A $\to$ B： m
2. B $\to$ A： m, N_B
3. A $\to$ B： $m, Sig_{S_A}(m, N_B, B)$

Applying the signature-based authenticator above to each flow in protocol 2 and joining (piggy-baking) the common flows， we get our new protocol in UM. According to Definition 1 and Theorems 1, protocol 1 is a SK-secure protocol in UM. Therefore, protocol 1 can provide the corresponding security attributes in CK, which is stronger than that in BR93. Furthermore, the colluding attack problem does not exist in protocol 1, since the TA's don't know the private keys of the parties.

4.3 SAKA

Current HTTP digest based authentication mechanism in SIP is vulnerable to many forms of attacks, such as man-in-the-middle, server spoofing and off-line password guessing attacks[5] etc, due to the lack of securing all headers and parameters in SIP which would possibly need protection.

After presenting with the key agreement protocol provably secure in CK model, we apply it to SIP environment and introduce a new secure authentication and key agreement mechanism as shown in Fig.2. SAKA doesn't need a pre-share password, since it is based on CL-PKC. When a user requests to access the resource of the server, or call some other user, he proceeds with the following steps:

Step 1: entity *A* chooses a random number $a \in_R \mathbb{Z}_q^*$, compute $T_A = aP$. Then *A* sends a REQUEST to entity *B* including his SIP identity ID_A, public key P_A, *s* and $T_A = aP$.

Step 2: on receiving the REQUEST, entity *B* chooses a random number $b \in_R \mathbb{Z}_q^*$, compute *nonce*=*H* (*realm*, *time*), $T_B = bP$ and *Sign*(*nonce*, *realm*, ID_A, T_A, T_B, P_B), and then *B* sends a CHALLENGE to entity *A* including *nonce*, *realm*, $T_B = bP$, *Sign*(*nonce, realm*, ID_A, T_A, T_B, P_B).

Step 3: on receiving the CHALLENGE, entity *A* compute the *nonce* first according to the *realm* and *time*, if the *nonce* had been used, *A* halts this protocol run; or else *A* verifies *B*'s signature first, if it is valid, he computes the response *Sign*(*nonce*, *realm*, ID_A, T_A, T_B, P_A) using his own private key, and then sends the RESPONSE to *B*. At the same time, if the signature is valid, *A* authenticates the identity of entity *B*, and he can compute K_{AB}= $\hat{e}(S_A, T_B) \cdot \hat{e}(Q_B, aY_B)$, and gets the shared key $SK_{AB}=H(K_{AB})$. The role of *nonce* is as a session identifier to resist replay attack.

Step 4: after the RESPONSE is received, entity *B* verifies the signature using P_A, if succeeds, *B* authenticates entity *A*'s identity, and he computes K_{BA}= $\hat{e}(S_B, T_B) \cdot \hat{e}(Q_A, aY_A)$ and the shared key $SK_{BA}=H(K_{BA})$= $SK_{AB}=H(K_{AB})$.

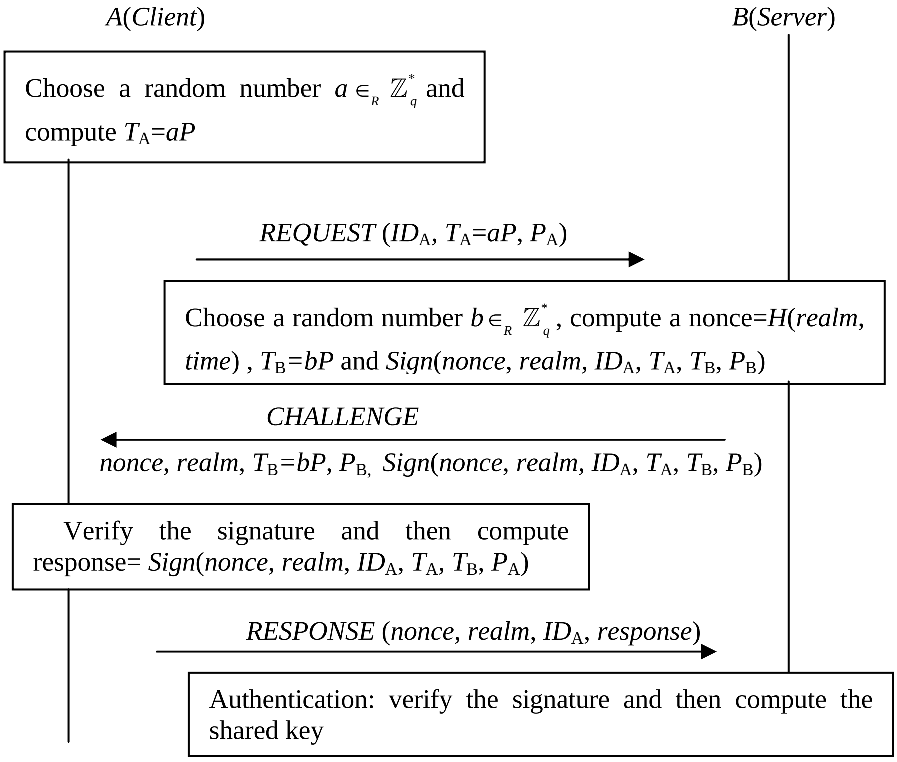
A(Client)
B(Server)
Choose a random number $a \in_R \mathbb{Z}_q^*$ and compute $T_A=aP$
REQUEST (ID_A, $T_A=aP$, P_A)
Choose a random number $b \in_R \mathbb{Z}_q^*$, compute a nonce=H(realm, time), $T_B=bP$ and Sign(nonce, realm, ID_A, T_A, T_B, P_B)
CHALLENGE
nonce, realm, $T_B=bP$, P_B, Sign(nonce, realm, ID_A, T_A, T_B, P_B)
Verify the signature and then compute response= Sign(nonce, realm, ID_A, T_A, T_B, P_A)
RESPONSE (nonce, realm, ID_A, response)
Authentication: verify the signature and then compute the shared key

Fig.2. SAKA：Secure authentication and key agreement mechanism for SIP.

Obviously, the handshake process of SAKA is based on the challenge response handshake of Digest, and thus it can operate without changing the semantics of RFC2617 HTTP authentication. Additionally, for the case of REGISTER, keying material $T_A=aP$, $T_B=bP$ and the computation of session key can be omitted.

4.3.1 Assumptions

To ensure that SAKA operates normally and securely, it is necessary to assume that the BDH parameters are generated and agreed on at the beginning of the system initiation and publicly available to all parties, the master key of the KGC is assumed to be private and secure, and the nonce will be a function of the realm and time, and will not be reused [9]. Furthermore, it is assumed that the KGC's are accessible by all entities in the system.

4.3.2 Security Attributes

SAKA is provably secure in CK security model, and thus it provides the following security attributes:

(1). **Mutual authentication**: as the description mentioned above, after a normal running of the protocol, the two participants authenticate each other.

(2). **Implicated key confirm**: if both of the signatures are verified to be valid, both participants can be sure that they are holding a special shared key between them.

(3). **Perfect forward secrecy**: the compromise of long-term private key doesn't affect the security of forward shared keys, since parameter *a*, *b* were erased after the protocol run.

Furthermore, we list out the main attacks which our new scheme can resist and corresponding security attributes or configurations to achieve this in table.1.

Table 1. Attacks and security attributes or configurations.

Anti-attacks	Security attributes or configurations
Man-in-the-middle attack	Mutual authentication
Session hijack attack	Mutual authentication
Server spoofing attack	Mutual authentication
Replay attack	Using nonce for freshness
Caller-ID impersonation attack	Using SIP identity to construct public key

5 Discussion

It is reasonable to insert two attributes for carrying key agreement messages and signatures, since additional attributes for describing session attributes are allowed to insert into the SDP messages as defined in RFC2327.

As stated in previous section, the current authentication mechanism is vulnerable to all the attacks listed in the above table, and a key agreement protocol is required by SRTP in SIP. Our SAKA mechanism solves out these problems, achieving the authentication and establishing a shared secret between previously unknown parties.

On one hand, since SAKA is based on certificateless public key cryptography, a large scale of PKI and the CA's, which are very expensive and needed by original authentication mechanism and [4~7], are avoided. The Caller-ID impersonation attack is also invalid in SAKA, since the public key is constructed using user's SIP identity. At meantime, the key escrow, the colluding attack and peer-to-peer connections problems in [9] are also solved out, for the following reasons:

(1). **Key escrow and colluding attack**: in SAKA (based on certificateless public key cryptography), KGC's don't know the private keys of participants, and even if the KGC's collude, nothing meaningful can be revealed.

(2). **Supports peer-to-peer connections**: since KGC cooperates with the parties to generate private keys and the identity of each party is bind to his public key, two parties can contact directly, irrespective of their various security domains.

One the other hand, SAKA is provably secure in CK security model, and thus it provides the corresponding security attributes: mutual authentication, implicated key confirm, perfect forward secrecy etc. Due to these security attributes, SAKA is immune to the main attacks suffered in current authentication mechanism in SIP: man-in-the-middle attack, session hijack attack and server spoofing attack.

Furthermore, SAKA doesn’t need a pre-share password between client and server, which is of great meaning when considering scalable, and the use of nonce provides SAKA with immunity to replay attack.

However, there also exist some limitations in SAKA, just as some of the limitations in [9], which need our further research:

(1). The use of SAKA when more than two parties are present in a call has not been investigated;

(2). There is a potential vulnerability in the new scheme when access to the PSTN is required as the authentication and key agreement is being performed by the media Gateway, not the PSTN user, therefore, no assurance of identity can be given to either party.

Additionally, since SAKA is based on CL-PKC, the computation of elliptic curve pairings is unavoidable, which is considerably expensive. The authentication and key agreement process involves two elliptic curve pairings, two signatures and two hashs. The cost is the same to both initiator and responder, since the process is symmetric.

6 Conclusion

This paper proposes a new authentication and key agreement mechanism using CL-PKC, which achieves mutual authentication, and a shared secret between previously unknown parties is established. Security of SAKA is proved in CK security model, which is stronger than that proved in BR93 model, and then security attributes and the ability of resisting the main attacks suffered in current HTTP digest based authentication in SIP are given. Furthermore, we discuss the advantages of SAKA in two aspects by comparing with the current solutions. Finally, the limitations are listed out, and we need do further research to improve them.

References

[1]. J. Rosenberg et al. SIP: Session Initiation Protocol. IETF RFC 3261, 2002

[2]. S. Al-Riyami and K. Paterson. Certificateless public key cryptography, Advances in Cryptology-Asiacrypt’2003, Lecture Notes in Computer Science, vol.2894, pp.452-473.

[3]. J. Franks, P. Hallam-Baker, J. Hostertler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP Authentication: Basic and Digest Access Authentication. Request for comments 2617, Internet Engineering Task Force, 1999.

[4]. Salsano Stefano, Veltri Luca and Papalilo Donald. SIP security issues: the SIP authentication procedure and its processing load. IEEE Network, Volume 16, Issue 6, (2002) pp.38-44.

[5]. C.C. Yang, R.C. Wang, W.T. Liu. Secure authentication scheme for session initiation protocol, Computer &Security, vol.24, (2005) pp.381-386.

[6]. R. Srinivasan, V. Vaidehi, K. Harish, K. Lakshmi-Narasimhan, S. LokeshwerBabu and V. Srikanth. Authentication of Signalling in VoIP Applications. Communications, Asia-Pacific Conference, (2005) pp.530-533.

[7]. L. Kong, V.B. Balasubramaniyan and M. Ahamad. A lightweight scheme for securely and reliably locating SIP users. VoIP Management and Security, IEEE Workshop, (2006) pp.9-17.

[8]. P. Vesterinen. User authentication in SIP, TKK T-110.5290 seminar on Network Security, 12-11/12, 2006.

[9]. Jared Ring, Kim-Kwang Raymond Choo, Ernest Foo and Mark Looi. A New authentication Mechanism and Key Agreement Protocol for SIP Using Identity-based Cryptography, Proceedings of AusCERT R&D Stream, (2006) pp.61-72.

[10]. R. Canetti, H. Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann ed.Proceedings of Eurocrpt'01. Lecture Notes in Computer Science, vol.2045, pp.453-474.

[11]. M. Bellare and P. Rogaway. Entity authentication and key distribution. In Advances in Cryptology- CRYPTO’93, Lecture Notes in Computer Science, vol.773, pp.232-249.

[12]. W.B. Mao. Modern Cryptography: Theory and Practice. Publishing House of Electronics Industry, pp.397-401, 2004.

[13]. M. Baugher, D. McGew, M. Nasland, E. Carrara, and K. Norman. The Secure Real-time Transport Protocol (SRTP). Request For Comments 3711, Internet Engineering Task Force, 2004.