Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 19(1): 66–85.
DOI: 10.3934/mbe.2022003
Received: 22 July 2021
Accepted: 17 September 2021
Published: 08 November 2021

Research article

An anonymous SIP authenticated key agreement protocol based on elliptic curve cryptography

Yanrong Lu[1,*] **and Dawei Zhao**[2,*]

[1] School of Safety Science and Engineering, Civil Aviation University of China, Tianjin, China

[2] Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan, China

* **Correspondence:** Email: yr_lu@cauc.edu.cn, zhaodw@sdas.org.

Abstract: Designing a secure authentication scheme for session initial protocol (SIP) over internet protocol (VoIP) networks remains challenging. In this paper, we revisit the protocol of Zhang, Tang and Zhu (2015) and reveal that the protocol is vulnerable to key-compromise impersonation attacks. We then propose a SIP authenticated key agreement protocol (AKAP) using elliptic curve cryptography (ECC). We demonstrate the correctness of the protocol using Burrows-Abadi-Needham (BAN), and its security using the AVISPA simulation tool. We also evaluate its performance against those of Zhang, Tang and Zhu, and others.

Keywords: authentication; session initial protocol; security; privacy; elliptic curve cryptography

1. Introduction

With the rapid growing of wireless networks, there is a corresponding increase in demand for multimedia-supported services, such as internet protocol television (IPTV) [1, 2] video conference, and 3D holographic displays. Such services can be offered via voice over IP (VoIP) [2, 3] services using session initial protocol (SIP) [3–7], a text-based signalling protocol. SIP has also been deployed for IP multimedia implementations [8, 9], smart home and network management [10, 11] and mobility management [12–14].

There are, however, open security and privacy challenges when transmitting voice packets over an open network. For example, designing a provably secure and efficient authentication protocol for SIP remains a challenging task. The original authentication mechanism in SIP was based on hypertext transport protocol (HTTP) [15] digested authentication, designed to provide only data authentication. Yang, Wang and Liu [16] introduced the first Diffie-Hellman authenticate key agreement protocol

(AKAP) for SIP using the client-server model. However, it was later found to be vulnerable to server spoofing and off-line password guessing attacks [16]. Several elliptic curve based SIP AKAP has also been presented in the literature [17–24], due to the advantages elliptic curve based protocols offered over Diffie-Hellman based protocols (e.g., reduced computational and storage costs). For example, the key length for elliptic curve cryptography (ECC) is much smaller than RSA and ElGamal cryptosystem at the same level of security. Similar to the troubled history of Diffie-Hellman-based protocols, several published ECC-based protocols were found to be insecure after their publication. For example, the protocol of Wu, Zhang and Wang [25] was found to be vulnerable to a range of attacks in [26]. An improved protocol was then presented.

To minimize the impact of a security breach at the server, one of the many desired security features in SIP authentication is to ensure that users' passwords (which may be stored as plaintext) are securely stored. To withstand stolen verifiers attack at the SIP server, a number of smart card-based AKAPs were proposed in the literature, and some of them were subsequently found to be flawed [24, 27–31].

More recently in 2016, Zhang, Tang and Zhu [32] introduced an energy-efficient AKAP for SIP. In this paper, we demonstrate that the proposed protocol is vulnerable to key-compromise impersonation attacks, in violation of their security claims. We then propose an ECC-based AKAP, and demonstrate its correctness and security respectively using BAN logic and the Automated Validation of Internet Security Protocols and Applications (AVISPA) simulation tool [33,34]. We also evaluate the performance of the proposed protocol.

2. Preliminaries

The basic knowledge on ECC and some notations are introduced as follows [35].

Definition 1. $\mathbf{E}_p(a, b) : y^2 = (x^3 + ax + b) mod p$ be the form of an elliptic curve $\mathbf{E}_p(a, b)$ over $\mathbf{F}_p$ with $a, b \in \mathbf{F}_p$ satisfying $(4a^3 + 27b) mod p \neq 0$.

Assumption 1. Elliptic curve discrete logarithm problem (ECDLP): Known two points aP and P over $\mathbf{E}_p(a, b)$, to determine the random number $a \in \mathbf{Z}_q^*$. It is a hard problem in polynomial time such that the security is achieved.

Assumption 2. Elliptic curve Diffie-Hellman problem (ECDHP): Known two points aP and bP over $\mathbf{E}_p(a, b)$, to calculate the point abP, where the random numbers $a, b \in \mathbf{Z}_q^*$. abP can not be solved with non-negligible probability in polynomial time.

3. Review of Zhang, Tang and Zhu's protocol

Zhang, Tang and Zhu's protocol consists of four phases: initiation, registration, authentication, and password change.

3.1. Initiation

S selects its private key as s, gets its public key as sP. Also, S releases the $\{P_{pub}, P, h(), E(\mathbf{F}_p)\}$, where $h()$ be the hash function.

3.2. Registration

Step 1. U_i computes $C_1 = h(PW_i \oplus r)$ and sends $\{ID_i, C_1\}$ to S via a secure channel. In the formula, r is a random number.

Step 2. S calculates $C_3 = h(ID_i \oplus s) \oplus C_1$ and sends back a smart card including $\{C_3\}$ to U_i.

Step 3. U_i writes r into the smart card, which has stored the values $\{C_3, r, h()\}$.

3.3. Authentication

Step 1. The smart card asks U_i to enter: identity ID_i and password PW_i. Based on the two values, the smart card derives $h(ID_i \oplus s)$ by computing $C_3 \oplus h(ID_i \oplus PW_i)$. Subsequently, the card picks two random numbers r_1, r_2, and computes $C_4 = r_1 P$, $C_5 = r_1 C_2 P_{pub}$, and $C_6 = h(C_5) \oplus (h(ID_i \oplus s) \oplus r_2, (C_5)_x, (C_5)_y)$. In the formula, $(C_5)_x$ and $(C_5)_y$ are x- and y-coordinate values of point C_5. Then, U_i delivers a request message REQUEST $\{ID_i, C_4, C_6\}$ to S.

Step 2. S calculates $C_2 = h(ID_i \oplus s)$ and retrieves $(h(ID_i \oplus s) \oplus r_2, (C_5)_x, (C_5)_y)$ by computing $h(sC_2C_4) \oplus C_6$. Then S checks whether $(C_5)_x, (C_6)_y \stackrel{?}{=} (sC_2C_4)_x$, $(sC_2C_4)_x$. If this holds, S derives r_2 by means of computing $C_2 \oplus h(ID_i \oplus s) \oplus r_2$. Then, S picks two random numbers r_3, r_4 and computes $C_7 = r_3 P$, $SK = h(C_4, r_3 C_4, C_7)$, and $Auth_s = h(h(ID_i \oplus s), r_2, (SK)_x, (C_5)_x, (SK)_y, (C_5)_y)$. Next, S replays a challenge message CHALLENGE $\{realm, C_7, Auth_s, r_4\}$.

Step 3. U computes $SK = h(C_4, r_1 C_7, C_7)$ and verifies whether $h(C_2, r_2, (SK)_x, (C_5)_x, (SK)_y, (C_5)_y) \stackrel{?}{=} Auth_s$. If this holds, U computes $Auth_u = h((SK)_x, (r_4 + 1), (SK)_y)$ and transmits a response message REPONSE $\{realm, Auth_u\}$ to S.

Step 4. Once S verifies whether $h((SK)_x, (r_4 + 1), (SK)_y) \stackrel{?}{=} Auth_u$. If the equation is correct, S and U successfully shares a common session key $SK = r_1 r_3 P$.

3.4. Password change

When U_i plans to change a password into a new one, he needs a helping of S. The process is as follows.

Step 1. U_i keys his old password PW_i and retrieves $Z = h(ID_i \oplus s)$ by computing $h(PW_i \oplus r) \oplus C_3$ and $V = Enc_{(SK)_x}(h(PW_i^* \oplus r^*), ID_i, R, Z)$. In the formula, PW_i^* and r^* are the new password and the random number. Next, U_i submits $\{V\}$ to S.

Step 2. S computes $h(ID_i \oplus s)$, and and examines whether it is equivalent to the decrypted value Z from V, using x-coordinate of point SK. Subsequently, S computes $C_3^* = h(PW_i^* \oplus r^*) \oplus h(ID_i \oplus s)$ and $W = Enc_{(SK)_x}(C_3^*, h(C_3^*, (R + 1)))$. S then returns $\{W\}$ to U_i.

Step 3. U_i checks whether $h(C_3^*, R + 1)$ is equivalent to the derived one which comes from received message W. If it is true, U_i uses the values $\{C_3^*, r^*\}$ instead of the old one.

4. Cryptanalysis of Zhang, Tang and Zhu's protocol

Before performing security analysis on the protocol of Zhang, Tang and Zhu, we assume that $\mathbb{A}$ could intercept any packets which are delivered in the authentication phase. Also, assume $\mathbb{A}$ can perform the corresponding computation.

Resilience to key compromise impersonation (KCI) attacked is considered as an important, and a desired security attributed to a key exchange protocol. It means an adversary $\mathbb{A}$ once has obtained one party's long-term private key. The saboteur may disguise another entity to the corrupted one. However, the three-round protocol of Zhang, Tang, and Zhu does not provide KCI-resilient. We show more detail on how $\mathbb{A}$ launches such an attack and achieves its goal. It breaks the confidentiality of the session key established.

Step 1. $\mathbb{A}$ compromises the perpetual secret key s of the server S. And $\mathbb{A}$ obtains the user's identity ID_i utilizing intercepting the communication messages;

Step 2. Based on the two parameters, $\mathbb{A}$ calculates $C_2 = h(ID_i \oplus s)$ and then generates two random numbers $r_1^{'}$, $r_2^{'}$;

Step 3. $\mathbb{A}$ computes $C_4 = r_1 P$, $C_5 = r_1^{'} C_2 P_{pub}$ and $C_6 = h(C_5) \oplus (h(ID_i \oplus s) \oplus r_2^{'}, (C_5)_x, (C_5)_y)$ and transmits the REQUEST message $\{ID_i, C_4, C_6\}$ to S;

Step 4. S calculates C_2 and verifies $(C_s 5)_x, (C_5)_y \stackrel{?}{=} (sC_2C_4)_x, (sC_2C_4)_y$. If this holds, S computes $C_7 = r_3 P$, $SK = h(C_4, r_3 C_4, C_7)$, and $Auth_s = h(h(ID_i \oplus s), r_2^{'}, (SK)_x,$
$(C_5)_x, (SK)_y, (C_5)_y)$. In the formula, r_3 and r_4 are the random numbers. Next, S transmits the CHALLENGE message $(realm, C_7, Auths, r_4)$ to U_i;

Step 5. $\mathbb{A}$ calculates $SK = h(C_4, r_1 C_7, C_7)$ and verifies whether $h(C_2, r_2, (SK)_x,$
$(C_5)_x, (SK)_y, (C_5)_y) \stackrel{?}{=} Auth_s$. Also, $\mathbb{A}$ computes $Auth_u = h((SK)_x, (r_4 + 1), (SK)_y)$ and sends the RESPONSE message $(realm, Auth_u)$ to S.

Step 6. S needs to check whether $h((SK)_x, (r_4 + 1), (SK)_y)$ is equivalent to the received $Auth_u$. If it is false, S discards the packets; Otherwise, S accepts $\mathbb{A}$, and agrees on the session key $SK = r_1^{'} r_3 P$ as their subsequent shared key.

More seriously, $\mathbb{A}$ could even guess the correct password PW_i once the secret key of S is compromised. $\mathbb{A}$ is by checking whether $h(PW_i^* \oplus r) \oplus h(ID_i \oplus s)$ until the equation holds. In this formula, PW_i^* is an arbitrary element that is selected from the password candidate set.

5. The proposed protocol

To solve the security problems found in Zhang, Tang and Zhu's protocol, we develop a secure SIP authentication protocol with a key agreement facility. Unlike Zhang, Tang and Zhu's design, our proposal comprises of five phases: initialization, registration (Table 1), login (Table 2), key agreement (Table 3) and password update, where the password update process is concise and convenient, that lying in it does not require interaction with the server. The proposed protocol is explained below along with Figure 1.

5.1. Initialization

S selects an additive group G, with a generator P of large prime order q, including points of an elliptic curve E over a finite field $\mathbf{F}_p$. S publishes the public parameters $\{h(), sP, P, E, q\}$, where $s \in \mathbf{F}_p, P \in \mathbf{E}_p(a, b)$.

5.2. Registration

Once a new user U_i attempts to access services, he first selects his identity ID_i and password PW_i of his/her choice.

Step 1. The user U_i picks out a random number r, and calculates $h(PW_i, r)$. U_i then transmits the registration request message $\{ID_i\}$ to the proxy server, through a private channel.

Step 2. The proxy sever S calculates $X_1 = h(ID_i, s)P$ and $X_2 = h(ID_i)nsP$. In the formula, n is a random number. Next, S personalizes U_i's smart card which remains the values $\{X_1, X_2\}$ to U_i via a secure channel.

Step 3. U_i computes $X_3 = h(PW_i, r)h(ID_i, s)P$ and $X_4 = h(h(ID_i) \oplus h(PW_i, r))$. U_i finally stores r into his smart card. Note that the smart card includes the information $\{r, X_2, h(), X_3, X_4\}$.

Table 1. Algorithm 1: Registration.

Algorithm 1: Registration
Input: ID_i, PW_i, r
Output: smart card
1: Select r,
2: Compute $h(PW_i, r)$,
3: Transmit ID_i and $h(PW_i, r)$ to S.
4: Select n, $X_1 = h(ID_i, s)^{-1}P$, $X_2 = h(ID_i)nsP$
5: Transmit X_1 and X_2 to U.
6: Compute $X_3 = h(PW_i, r)h(ID_i, s)^{-1}P$,
7: $X_4 = h(h(ID_i) \oplus h(PW_i, r)$,
8: Smart card $\{r, X_3, h(), X_4, X_5\}$.

5.3. Login

U_i enters his identity ID_i and password PW_i after putting the smart card into the card slot.

Step 1. The smart card verifies whether the condition $h(h(ID_i) \oplus h(PW_i, r)) \stackrel{?}{=} X_4$ holds.

Step 2. If the equality holds, the smart card then gets $h(ID_i, s)P$ by calculating $h(PW_i, r)^{-1}X_3$ and $nsP = h(ID_i)^{-1}X_2$. Also, a random number r_1 is generated, the value $X_5 = Enc_{nsP}(ID_i, r_1P, sP)$ to verify.

Step 3. U_i delivers the login request message REQUEST $\{X_5\}$ to S through a public path.

5.4. Key agreement

Step 1. S uses its private key s and secrets parameter n to retrieve (ID_i, r_1P, sP).

Step 2. S proceeds to generate two random numbers r_2, r_3, and calculates the temporary key $SK = h(r_2r_1P, ID_i)$ to be shared with the user U_i. Also, the smart card calculates $X_6 = Enc_{h(ID_i,s)P}(r_2P, r_1P)$ and $Auth_s = h(SK, r_1P, r_3)$. And then the smart card responds with the challenge message CHALLENGE $\{realm, X_6, Auth_s, r_3\}$ to U_i.

Step 3. The smart card gets the values (r_2P, r_1P) by decrypting X_6. And then, the temporary key $SK = h(r_2r_1P, ID_i)$ is gotten by U_i and to be shared with the server S. Verifying whether $h(SK, r_1P, r_3) \stackrel{?}{=} Auth_s$. If the condition holds, U_i deems S as the legitimate server.

Step 4. The smart card computes $Auth_u = h(SK, r_2P, r_3 + 1)$, and transmits a response message RESPONSE $\{realm, Auth_u\}$ to S through a public channel.

Table 2. Algorithm 2: Login.

Algorithm 2: Login
Input: ID_i, PW_i, smart card
Output: **true**: output X_5; **false**: failure
1: **if** $h(h(ID_i) \oplus h(PW_i, r)) \stackrel{?}{=} X_4$ **then**
2: Select r_1,
3: $h(ID_i, s) = h(PW_i, r)X_3^{-1}P$,
4: $nsP = h(ID_i)^{-1}X_2$,$X_5 = Enc_{nsP}(ID_i, r_1P)$.
5: return **true**
6: **else**
7: return **false**
8: **end if**

Step 5. S examines the verification condition $h(SK, r_2P, r_3 + 1) \stackrel{?}{=} Auth_u$. If this equation holds, S ensures U_i as authentic and agrees on the session key SK as valid key.

Table 3. Algorithm 3: Key agreement.

Algorithm 3: Key agreement
Input: ID_i, PW_i, X_5
Output: **true**: success; **false**: failure
1: $(ID_i, r_iP) \leftarrow Dec_{nsP}(X_5)$,
2: Select r_2,r_3
3: Compute $SK = h(r_2r_1P, ID_i)$,
4: $X_6 = h(ID_i, s)r_2P$,$Auth_s = h(SK, r_2P, r_3)$.
5: Transmit X_6, $Auth_s$ and r_3 to U_i.
6: Compute $r_2P = h(ID_i, s)^{-1}X_6$,$SK = h(r_1r_2P, ID_i)$.
7: **if** $Auth_s \stackrel{?}{=} h(SK, r_1P, r_3)$ **then**
8: $Auth_u = h(SK, r_2P, r_3 + 1)$.
9: Transmit $Auth_u$ to S
10: **if** $Auth_u \stackrel{?}{h} (SK, r_2P, r_3 + 1)$ **then**
11: return **true**
12: **else**
13: return **false**
14: **else**
15: return **false**
16: **end if**

5.5. Password updating

The following mechanism achieves altering the password of a legal user U_i without interacting S.

Step 1. U_i puts his smart card into the card slot and waits for commands of the terminal to provide the identity ID_i and password PW_i. The smart card verifies that if the condition $h(h(ID_i) \oplus h(PW_i, r)) \stackrel{?}{=} X_4$ holds. If the validation does not validate, the session is quitted promptly. Otherwise, the smart card derives $h(ID_i, s)P$ by computing $X_3 h(PW_i, r)^{-1}$, and requests a new password.

Step 2. U_i picks his new password PW_i^* and the random number r^*. The smart card calculates $X_3^* = h(PW_i^*, r^*)h(ID_i, s)P$ and $X_4^* = h(h(ID_i) \oplus h(PW_i^*, r^*))$.

Step 3. The smart card discards X_3 and X_4 but keeps X_3^* and X_4^* in its memory for renewal.

6. Security analysis of the proposed protocol

We confirm our proposal could achieve a mutual handshake using well-popular BAN logic [36]. Also, the robustness of our proposal is validated via the universally applicable simulation tool-AVISPA [33, 34]. In addition, we provide informal cryptanalysis so as to demonstrate our proposal is well protecting against relevant security attacks.

6.1. Verification of the proposal under BAN Logic

BAN logic is a well-known formal method used to strictly prove the authentication protocols' security, or find security vulnerabilities. Subsequently, we will introduce more details about BAN logic. BAN logic includes basic logical notations and some logic postulates. According to these preliminaries, we show the desired goals, idealized form, assumptions for our protocol. And we finally demonstrate its correctness.

Notations

· $P| \equiv X$: P deems X is true;
· $P \triangleleft X$: P observes X;
· $P| \sim X$: P ever have sent X;
· $P \Rightarrow X$: P judges X;
· $\#X$: X is fresh;
· $P \stackrel{K}{\leftrightarrow} Q$: share a key K between P and Q;
(X, Y): X or Y is one portion of the formula (X, Y);
· $< X, Y >_K$: K is the key to encrypt X and Y.

Postulates

· Message-meaning rule: $\frac{A|\equiv A \stackrel{K}{\leftrightarrow} B,\ A \triangleleft <X>_K}{A|\equiv |B \sim X}$;
· Fresh conjuncatenation rule: $\frac{A|\equiv \#(X)}{A|\equiv \#(X,\ Y)}$;
· Belief rule: $\frac{A|\equiv X,\ A|\equiv Y}{A|\equiv (X,\ Y)}$;
· Nonce-verification rule: $\frac{A|\equiv \#(X),\ A|\equiv B|\sim X}{A|\equiv B|\equiv X}$;
· Jurisdiction rule: $\frac{A|\equiv B \Rightarrow X,\ A|\equiv B|\equiv X}{A|\equiv X}$.

Goals

· $Goal_1$: $S| \equiv ID_i$
· $Goal_2$: $S| \equiv U_i \stackrel{SK}{\longleftrightarrow} S$

· $Goal_3$: $S| \equiv U_i| \equiv U_i \overset{SK}{\longleftrightarrow} S$
· $Goal_4$: $U_i| \equiv U_i \overset{SK}{\longleftrightarrow} S$
· $Goal_5$: $U_i| \equiv S| \equiv U_i \overset{SK}{\longleftrightarrow} S$

Idealized form

· $U_i \rightarrow S$: $\{X_5, Auth_u\}$
X_5: $< ID_i, r_1P, sP >_{U_i \overset{nsP}{\longleftrightarrow} S}$,
$Auth_u$: $< r_2P, r_3 + 1, SK >_{U_i \overset{SK}{\longleftrightarrow} S}$;
· $S \rightarrow U_i$: $\{X_6, Auth_s\}$
X_6: $< r_2P, r_1P >_{U_i \overset{h(ID_i,s)}{\longleftrightarrow} S}$,
$Auth_s$: $< r_1P, r_3, SK >_{U_i \overset{SK}{\longleftrightarrow} S}$

Assumptions

· A_1: $U_i| \equiv ID_i$;
· A_2: $U_i| \equiv \#r_1$;
· A_3: $S| \equiv \#n$;
· A_4: $S| \equiv \#s$;
· A_5: $U_i| \equiv (U_i \overset{nsP}{\leftrightarrow} S)$;
· A_6: $S| \equiv (U_i \overset{nsP}{\leftrightarrow} S)$;
· A_7: $U_i| \equiv (U_i \overset{h(ID_i,s)}{\leftrightarrow} S)$;
· A_8: $S| \equiv U_i \Rightarrow ID_i$
· A_9: $S| \equiv U_i \Rightarrow r_1$
· A_{10}: $S| \equiv \#r_2$
· A_{11}: $U_i| \equiv S \Rightarrow r_2$
· A_{12}: $S| \equiv \#r_3$
· A_{13}: $U| \equiv \#r_3$

Proofs

· According to message X_5, we have
P_1. $S \triangleleft < ID_i, r_1P, sP >_{U_i \overset{nsP}{\longleftrightarrow} S}$
· By P_1, A_6 and message-meaning rule, we obtain
P_2. $S| \equiv U_i| \sim (ID_i, r_1P, sP)$
· From P_2, A_4 and fresh conjuncatenation rule, we derive
P_3. $S| \equiv \#(ID_i, r_1P, sP)$
· Since P_2, A_3 and nonce-verification rule, we get
P_4. $S| \equiv U_i| \equiv (ID_i, r_1P, sP)$
· According to P_4 and belief rule, we get
P_5. $S| \equiv U_i| \equiv ID_i$, $S| \equiv U_i| \equiv r_1P$
· From A_8, A_9 and jurisdiction rule, we derive
P_6. $S| \equiv r_1P$, $Goal_1$. $S| \equiv ID_i$
· Since $SK = h(r_1r_2P, ID_i)$, $Goal_1$, and P_6, we have
$Goal_2$. $S| \equiv U_i \overset{SK}{\longleftrightarrow} S$
· According to message $Auth_u$, we have
P_7. $S \triangleleft < r_2P, r_3 + 1, SK >_{U_i \overset{SK}{\longleftrightarrow} S}$

· By P_7, $Goal_2$ and message-meaning rule, we obtain

P_8. $S| \equiv U_i| \sim (r_2P, r_3 + 1, U_i \overset{SK}{\longleftrightarrow} S)$

· Since P_8, A_{10}, A_{11} and nonce-verification rule, we have

$Goal_3$. $S| \equiv U_i| \equiv U_i \overset{SK}{\longleftrightarrow} S$

· From message X_6, we attain

P_9. $U_i \triangleleft < r_2P, r_1P >_{U_i \overset{h(ID_i,s)}{\longleftrightarrow} S}$

· According to A_7, P_9 and message-meaning rule, we get

P_{10}. $U_i| \equiv S| \sim (r_2P, r_1P)$

· By A_2 and fresh conjuncatenation rule, we have

P_{11}. $U_i| \equiv \#(r_2P, r_1P)$

· According to P_{10}, P_{11} and nonce-verification rule, we attain

P_{12}. $U_i| \equiv S| \equiv r_2P$

· Since P_{12}, A_{11} and jurisdiction rule, we derive

P_{13}. $U_i| \equiv r_2P$

· From $SK = h(r_1r_2P, ID_i)$, A_1, A_2, P_{13}, we have

$Goal_4$. $U_i| \equiv U_i \overset{SK}{\longleftrightarrow} S$

· By message $Auth_s$, we attain

P_{14}. $U_i \triangleleft < r_1P, r_3, SK >_{U_i \overset{SK}{\longleftrightarrow} S}$

· Since P_{14}, $Goal_4$ and message-meaning rule, we have

P_{15}. $U_i| \equiv S| \sim (r_1P, r_3, U_i \overset{SK}{\longleftrightarrow} S)$

· According to P_{13}, P_{15}, $Goal_4$ and nonce-verification rule, we attain

$Goal_5$. $U_i| \equiv S| \equiv U_i \overset{SK}{\longleftrightarrow} S$

6.2. Formal security analysis

Theorem 1. The probability that an attacker $\mathbb{A}$ breaks the AKE security of our AKAP is

$Adv_P^{Ake}(\mathbb{A}) \leq \frac{q_h^2}{2^{l-1}} + \frac{2q_{send}}{|D|}$

where q_{send}, q_h and D denote the number of Send queries, Hash queries, and a uniformly distributed dictionary, respectively.

Proof: Game $G_i(i = 0, 1, 2)$ defines three games. Game G_0 is the factual attack, and game G_3 concludes a breach of the AKE security of our AKAP is asymptotically optima:

Game G_0: This game corresponds to the actual attack.

$Adv_P^{Ake}(\mathbb{A}) = |2Pr[Succ_0] - 1|$.

Game G_1: This game simulates the eavesdropping attack by querying $Execute(U^i, S^j)$ oracle, and then by querying $Test(P^i)$ oracle. It decides whether the result of *Test* is the real session key SK or a random value. We know that r_1P is derived by the server's secret key s, and secrets parameter n. That is, $\mathbb{A}$ has no way to compute r_1P through eavesdrop on the communication channel unless S is compromised. Also, r_2P is not impossible to obtain, unless it possesses both the smart card and password. Hence, intercepting is not probable for helping $\mathbb{A}$ to win in this game. Thus,

$Pr[Succ1] = Pr[Succ0]$.

Game G_2: This game models $Send(M, P^i)$ query, in which $\mathbb{A}$ can eavesdrop or alter the information from the transcripts. Then, games G_2 and G_1 are undistinguishable unless the collision occurring in

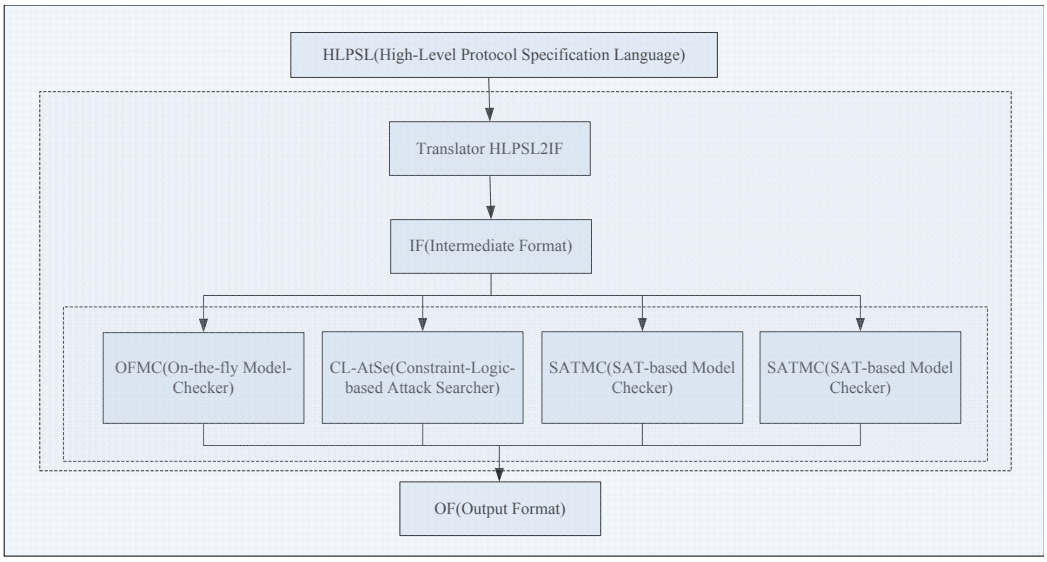

Figure 1. Architecture of the AVISPA tool.

G_2. Thus,

$|Pr[Succ2] - Pr[Succ1]| \leq q_h^2/2^l$.

Game G_3: This game models $Corrupt(SC)$ query, in which $\mathbb{A}$ has obtained the smart card to simulate the smart card breach attack. Since the password PW_i is protected by a cryptographic one-way function, where $X_3 = h(PW_i, r)h(ID_i, s)P$ and $X_4 = h(h(ID_i) \oplus h(PW_i, r))$. This implies that $\mathbb{A}$ has no way to check the password excepts possession of user's identity, or corrupts the server to get s. Hence, $|Pr[Succ3] - Pr[Succ1]| \leq q_{send}/|D|$.

6.3. Verifying protocol using AVISPA tool

AVISPA is a simulation engine for the automated validation of Internet security protocols and applications. Upon Dolev and Yao model, four model back-ends, called OFMC (On-the-fly Model-Checker), CL-AtSe (Constraint-Logic-based Attack Searcher), SATMC (SAT-based Model-Checker), and TA4SP (Tree Automata-based Protocol Analyzer) (Figure 1) are utilized for the validation using HLPSL (High-Level Protocol Specification Language). The HLPSL presentation of the protocol is compiled to IF by the translator-HLPSL2IF. IF is an entrance of the four different back-ends. The output OF is exported by using one of the four back-ends, which shows the conclusion if the AKAP is secure or insecure.

During the protocol execution, each entity act a role, which is a feature of AVISPA. We show the role specifications in HLPSL of Appendix the initiator, responder, session, and environment and goal in Appendix Figures 1–4. In our implementation, we assume S's private key is a public parameter. The privacy of three parameters and client-server authentication is verified:

· The secrecy_of subs1: the user confidential parameters ID_i is gotten only U_i and S.

· The secrecy_of subs2: the user confidential parameters PW_i is gotten only U_i.

· The secrecy_of subs3: the session key is gotten only U_i and S.

· Authenticaion_si_ui_auths: U_i validates S by receiving r_2 securely, r_2 is a ephemeral number of S.

· Authenticaion_si_ui_authu: U_i validates S by receiving r_1 securely, r_1 is a ephemeral number of S.

```
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL
PROTOCOL
  /opt/avispa-1.1/testsuite/results/LuP.if
GOAL
  As Specified
BACKEND
  CL-AtSe
STATISTICS
  Analysed   : 20 states
  Reachable  : 8 states
  Translation: 0.00 seconds
  Computation: 0.00 seconds
```

Figure 2. Simulation result in CL-AtSe model checker.

```
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /opt/avispa-1.1/testsuite/results/LuP.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.04s
  visitedNodes: 33 nodes
  depth: 6 plies
```

Figure 3. Simulation result in OFMC model checker.

After running the program under two back-ends CL-AtSe and OFMC, Figures 2 and 3 show that our proposal realizes the session security without imperfection.

6.4. Informal security analysis

We demonstrate that our proposal holds many security attributes, such as mutual authentication, anonymity, privileged insider attack, perfect forward secrecy, KCI-resistance, etc., under a condition. The condition is that $\mathbb{A}$ extracts all the data stored inside a user's smart card, or/and eavesdrops on all the messages involved in an authentication-key agreement session [37, 38].

6.4.1. Mutual authentication

Note that r_1P can be only decrypted by the legal server. Thus, after U_i receives CHALLENGE message depending upon the result of decryption test, U_i verifies the legitimacy of S by checking the

equivalence $h(SK, r_1P, r_3) \stackrel{?}{=} Auth_s$. Simultaneously, only the legal U_i can derive r_2P from CHALLENGE message. And hence, S assures that he is communicating with the legitimate U_i, employing checking the equivalence $h(SK, r_2P, r_3+1) \stackrel{?}{=} Auth_u$ after receiving RESPONSE message. Therefore, $\mathbb{A}$ cannot cheat any of the communicating entity. And thus the proposed protocol achieves proper mutual authentication.

6.4.2. Internal-privileged attack

U_i's password PW_i is hashed by the random number r during registration phase. And it is not delivered to S. Therefore, an honest but curious insider has no ability, to know the real password PW_i of U_i. In other words, the proposed protocol indicates good capability of defeating insider attack.

6.4.3. User anonymity

The identity ID_i of U_i is disguised for dual protection, involved in all the transmitted messages. For one thing, the random number n is picked excepts the private key s of S, based on the ECDLP assumption. For another, the plain-text ID_i is hashed by double times along with the secret parameters r_1P, and r_2P. And the two parameters are encrypted by nsP. That is, to identify two parameters are equate as determine ECDLP problem. The problem is one of the well-known difficult problems within polynomial time. In a word, the proposed protocol supports high user anonymity.

6.4.4. Perfect forward and backward secrecy

Suppose that perpetual privacy information s of S is compromised by $\mathbb{A}$, he is incapable of computing the current as well as the future session keys. It is noteworthy that the session key is related with three important parameters, i.e, U_i's identity, two random numbers r_1 and r_2 generated by U_i and S, which are present in the form of r_1P and r_2P, respectively. With the purpose of acquiring ID_i and r_1P from intercepted REQUEST message $\{X_5 = Enc_{nsP}(ID_i, r_1P, sP)\}$, another random number n is also needed. Unfortunately, no one but S knows what is the real random number n. More seriously, r_2P can not be derived through $X_6 = Enc_{h(ID_i,s)P}(r_1P, r_2P)$ without knowing ID_i of U_i. Next, assume that the current session key is corrupted by $\mathbb{A}$. He plans to deduce the next negotiatory key. The cause of failure of extraction the secret $SK = h(r_1P, r_2P, ID_i)$ lies in the irreversible property of hash function, he total has no way to compute the previous and future session keys thereby. In a conclusion, our protocol preserves perfect forward secrecy property.

6.4.5. Resist modification attacks

We assume $\mathbb{A}$ intercepts the REQUEST message $\{X_5\}$ submitted to S, he attempts to modify the parameter r_1P and transmit the forged message $\{X_5\}$ to S. However, he has no way to attain ID_i and n to compute the symmetric key nsP, and thus encrypt ID_i. Even with the smart card security breached, he is not strong enough to gain the exact values of ID_i, and s without the knowledge of PW_i. On the other hand, without knowing ID_i, n and s, it is impossible for $\mathbb{A}$ to learn r_1P picked by U_i intercepting CHALLENGE message. That is, $\mathbb{A}$ could neither impersonate as an authorised user nor masquerade as a legitimate sever, through eavesdropping the raw messages to tamper with them. In a word, our proposal is is resistance to modification attacks.

6.4.6. Resist off-line password attack

We suppose $\mathbb{A}$ gets $\{r, X_2, X_3, X_4, h()\}$ and all the public messages $\{X_5, X_6, Auth_s, r_3, Auth_u\}$ from the communication channel. First and foremost, the password PW_i of U_i do not involve in transcription between U_i and S. Hence, $\mathbb{A}$ has no way to ensure whether the guesses password is true or false. Secondly, to derive the correct password PW_i from X_3 is a computationally infeasible task for $\mathbb{A}$, in condition that he is unaware of the identity ID_i of U_i, and the private key s of S. Eventually, $\mathbb{A}$ is not entirely sure what are the real values of ID_i, and PW_i by X_4, because the two personal information are hidden in two-layer hash function. In conclusion, the proposed protocol could thwart off-line password attack successfully.

6.4.7. Resist key-compromise impersonation attack

The whole design presumes that the privacy information s is considered as an open parameter. In this case, $\mathbb{A}$ has no way to impersonate as an authorised user thereby accessing service resources. The resources are provided by the secure server. Let's analyze the cases. Aiming at playing a valid user, the identity of the real user U_i is urgent required, since the server will detect the attack while checking $Auth_u$. To get the real value of ID_i, another random number n, which only the legal server knows it, is also needed. Thus, it is not possible for $\mathbb{A}$ to try to impersonate as a legitimate user.

6.4.8. Session key agreement

Two parties independently negotiate an ephemeral session key $SK = h(r_1 r_2 P, ID_i)$. They keep communicate securely on the strength of it for a subsequent communication. Upon this ability, each entities can encrypt the following packets, to preserve the security of the handshake. Moreover, the negotiatory key is fresh for each session. The reason is that random numbers are different based on the property of hash function. As a consequence, deriving the session key through the eavesdropped information is a challenging task for $\mathbb{A}$.

6.4.9. Stolen-verifier attacks

The service provider does not create the password verification table of the service requester. Even though the service provider's database is available by $\mathbb{A}$, he still cannot steal and modify user passwords, and thus attain the authentication information of users. Thus it can be said that our proposal can withstand stolen-verifier attacks.

7. Performance considerations and functionality comparison

This part evaluates and examines the performance of our proposed scheme, and compare it with five related protocols relies basically on the ECC, one-way hash functions. In order to evaluate the entire computing cost for each protocol more accurately, the arithmetic mean for each cryptographic computation timings after running 1000 times are shown in Table 4. The processor is Intelr Pentiumr CPU G3250, 3.20GHz with 4.0GB of RAM running Windows 10. We use the jPBC library(2.0.0) [41], a Java port of the PBC library written in C [39, 40], the Java Development Kit used is the Oracle jdk 1.8.0 65. We used the Type A curves with the prime order q defined as $E(Fq) : y^2 = x^3 + x$, hash function as SHA-3 [19, 42], and symmetric encryption algorithm as AES [43]. The calculation expense and

Table 4. jPBC library primitives timings.

Operation	t_{htp}	t_{sm}	t_{pa}	t_h	t_{inv}	t_{sym}
Aritmetic mean	10.8966ms	10.5129ms	0.4338ms	0.0359ms	0.0428ms	0.1755ms

Note: t_{htp}: executing a hash to point operation; t_{sm}: executing an elliptic curve scalar multiplication; t_{pa}: executing an elliptic curve point addition; t_h: executing a hash function operation; t_{inv}: executing a modular inversion; t_{sym}: executing a symmetric encryption/decryption

Table 5. Computational cost comparison.

	Registration(ms)	Authentication(ms)
Ours	$t_{sm} + 5t_h + 3t_{htp}$ ≈48.3822	$4t_{sm} + 10t_h + 2t_{inv} + 2t_{sym}$ ≈43.8472
[32]	$2t_h$ ≈0.0718	$4t_{sm} + 9t_h + 2t_{htp}$ ≈64.1679
[30]	$2t_h + t_{inv}$ ≈0.1164	$6t_{sm} + 11t_h + 2t_{htp}$ ≈85.2655
[29]	$4t_h$ ≈0.1436	$6t_{sm} + 11t_h + 2t_{sym}$ ≈63.8233
[28]	$t_{sm} + 4t_h + t_{htp}$ ≈21.5531	$8t_{sm} + 11t_h + 2t_{htp} + 4t_{pa}$ ≈85.3511
[27]	$t_{sm} + 3t_h + t_{inv}$ ≈10.6634	$7t_{sm} + 12t_h$ ≈74.0211
[24]	$2t_{sm} + 2t_h + t_{pa}$ ≈21.5314	$7t_{sm} + 10t_h + t_{pa}$ ≈74.3831

execution time of the registration and key agreement phases with the revelent protocols [24, 27–30, 32] are listed in Table 5. Also, we compare the security attributes with the related protocols [24, 27–30, 32] (Table 6). The entire running time of our AKAP is lower than Tu et al. [24]. And the time is lower than Yeh, Chen and Shih's protocol [28] too. Table 5 has shown the results. However, one thing is ignored by these protocols [24, 27–30, 32] from Table 6. It is to analyze whether the design has ability to conquer key compromise impersonation attack. Additionally, the related protocols seem not to be considered as a SIP authenticated key agreement, which can be perfect or ideal. The reason is that the protocol lacks some essential security properties. In general, our proposal takes a better tradeoff between computational cost and security attributes, while comparing with the protocols [24, 27–30, 32].

Table 6. Security comparison.

	Ours	[32]	[30]	[29]	[28]	[27]	[24]
Mutual authentication	✓	✓	✓	✓	✓	✓	✓
Insider attack	✓	✓	✓	✓	✓	✓	✓
User anonymity	✓	–	✓	✓	✓	–	–
PFABS	✓	✓	✓	✓	✓	✓	✓
Modification attack	✓	✓	✓	✓	–	–	–
OLPG attack	✓	–	✓	✓	–	✓	✓
KCI attack	✓	–	–	–	–	–	–
SK agreement	✓	✓	✓	✓	✓	✓	✓
SV attack	✓	✓	✓	✓	✓	✓	✓
TS attack	✓	✓	✓	✓	✓	–	✓

Note: PFABS: perfect forward and backward secrecy; OLPG attack: off-line password guessing attack; SV attack: stolen-verifier attack; TS attack:time synchronization attack.

8. Conclusions

Despite the role that AKAP plays in ensuring the security of communication in an open network, designing secure and efficient protocols, including in a VoIP environment, remains challenging. For example, in this paper, we revisited and revealed vulnerabilities in the design of Zhang, Tang and Zhu's protocol [32]. We also presented an improved protocol and demonstrated its correctness and security, as well as demonstrating its utility in terms of performance efficiency. Future research will include implementing a prototype of the protocol and evaluating it in a real-world deployment.

Acknowledgments

The authors would like to thank all the editors and anonymous reviewers for their helpful advice. This paper is supported by the National Natural Science Foundation of China (No. 61802276), and the Fundamental Research Funds for the Central Universities of China (No.3122021027).

Conflict of interest

The authors declare that they have no known competing financial interest or personal relationship that could have appeared to influence the work reported in this paper.

References

1. C. E. Palau, J. Mares, B. Molina, M. Esteve, Wireless CDN video streaming architecture for IPTV, *Multimedia Tools Appl.*, **53** (2011), 591–613. doi: 10.1007/s11042-010-0516-0.
2. H. S. Fard, A. G. Rahbar, Physical constraint and load aware seamless handover for IPTV in wireless LANs, *Comput. Elec. Eng.*, **56** (2016), 222–242. doi: 10.1016/j.compeleceng.2016.01.005.
3. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, et al., SIP: Session initiation protocol, *RFC3261*, **2543** (2002), 1–151.
4. J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, et al., HTTP Authentication: basic and digest access authentication, *RFC2617*, **2617** (1999), 1–34.
5. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, et al., Hypertext transfer protocol – HTTP/1.1, *RFC2616*, **2068** (1997), 1–162.
6. H. Arshad, M. Nikooghadam, An efficient and secure authentication and key agreement scheme for session initiation protocol using ECC, *Multimedia Tools Appl.*, **1** (2016), 181–197. doi: 10.1007/s11042-014-2282-x.
7. M. Nikooghadam, H. Amintoosi, Perfect forward secrecy via an ECC-based authentication scheme for SIP in VoIP, *J. Supercomput.*, **76** (2020), 3086–3104. doi: 10.1007/s11227-019-03086-z.
8. C. Y. Chen, K. D. Chang, H. C. Chao, Transaction-pattern-based anomaly detection algorithm for IP multimedia subsystem, *IEEE Trans. Inf. Forensics Secur.*, **6** (2011), 152–161. doi: 10.1109/TIFS.2010.2095845.
9. Y. Zhang, X. Sun, B. Wang, Efficient algorithm for k-barrier coverage based on integer linear programming, *China Commun.*, **13** (2016), 16–23. doi: 10.1109/CC.2016.7559071.

10. W. E. Chen, Y. L. Huang, H. C. Chao, NAT traversing solutions for SIP applications, *Eur. J. Wireless Commun. Networking*, **2008** (2008), 639528. doi: 10.1155/2008/639528.
11. T. Ma, J. Zhou, M. Tang, Y. Tian, A. Al-Dhelaan, M. Al-Rodhaan, et al., Social network and tag sources based augmenting collaborative recommender system, *IEICE Trans. Inf. Syst.*, **98** (2015), 902–910. doi: 10.1587/transinf.2014EDP7283.
12. C. M. Huang, C. W. Lin, C. C. Yang, Mobility management for video streaming on heterogeneous networks, *IEEE MultiMedia*, **17** (2010), 35–35. doi: 10.1109/MMUL.2010.17.
13. T. Wu, R. Jhang, H. Chao, Efficient architecture and handoff strategy used for VoIP Sessions in SIP based wireless networks, *Wireless Pers. Commun.*, **43** (2007), 201–214. doi: 10.1007/s11277-006-9218-3.
14. Y. Lu, G. Xu, L. Li, Y. Yang Anonymous three-factor authenticated key agreement for wireless sensor networks, *Wireless Networks*, **25** (2019), 1461–1475. doi: 10.1007/s11276-017-1604-0.
15. E. Wilde, *Hypertext Transfer Protocol (HTTP)*, Springer, 1999.
16. C. C. Yang, R. C. Wang, W. T. Liu, Secure authentication scheme for session initiation protocol, *Comput. Secur.*, **24** (2005), 381–386. doi: 10.1016/j.cose.2004.10.007.
17. A. Durlanik, I. Sogukpinar, SIP authentication scheme using ECDH, *World Enformatika Soc. Trans. Eng. Comput. Technol.*, **1** (2007), 2659–2662.
18. N. Koblitz, A. Menezes, S. Vanstone, The state of elliptic curve cryptography, *Designs Codes Cryptography*, **19** (2000), 173–193. doi: 10.1023/A:1008354106356.
19. A. J. Menezes, S. A. Vanstone, P. C. Van Oorschot, *Handbook of Applied Cryptography*, CRC Press, 1996.
20. V. S. Miller, Use of Elliptic Curves in Cryptography, in *Advances in cryptology—CRYPTO 85* (ed. Hugh C Williams), Springer-VerlagBerlin, Heidelberg, (1985), 417–426.
21. R. Arshad, N. Ikram, Elliptic curve cryptography based mutual authentication scheme for session initiation protocol, *Multimedia Tools Appl.*, **66** (2013), 165–178. doi: 10.1007/s11042-011-0787-0.
22. D. He, J. Chen, Y. Chen, A secure mutual authentication scheme for session initiation protocol using elliptic curve cryptography, *Secur. Commun. Networks*, **5** (2012), 1423–1429. doi: 10.1002/sec.506.
23. Y. Lu, L. Li, H. Peng, Y. Yang, An anonymous two-factor authenticated key agreement scheme for session initiation protocol using elliptic curve cryptography, *Multimedia Tools Appl.*, **76** (2017), 1801–1815. doi: 10.1007/s11042-015-3166-4.
24. H. Tu, N. Kumar, N. Chilamkurti, S. Rho, An improved authentication protocol for session initiation protocol using smart card, *Peer Peer Networking Appl.*, **8** (2015), 903–910. doi: 10.1007/s12083-014-0248-4.
25. L. Wu, Y. Zhang, F. Wang, A new provably secure authentication and key agreement protocol for SIP using ECC, *Comput. Stand. Interfaces*, **31** (2009), 286–291. doi: 10.1016/j.csi.2008.01.002.
26. E. J. Yoon, K. Y. Yoo, C. Kim, Y. S. Hong, M. Jo, H. Chen, A secure and efficient SIP authentication scheme for converged VoIP networks, *Comput. Commun.*, **33** (2010), 1674–1681. doi: 10.1016/j.comcom.2010.03.026.

27. A. Irshad, M. Sher, E. Rehman, S. A. Ch, M. U. Hassan, A. Ghani, A single round-trip SIP authentication scheme for voice over Internet protocol using smart card, *Multimedia Tools Appl.*, **74** (2015), 3967–3984. doi: 10.1007/s11042-013-1807-z.
28. H. L. Yeh, T. H. Chen, W. K. Shih, Robust smart card secured authentication scheme on SIP using Elliptic Curve Cryptography, *Comput. Stand. Interfaces*, **36** (2014), 397–402. doi 10.1016/j.csi.2013.08.010.
29. M. S. Farash, S. Kumari, M. Bakhtiari, Cryptanalysis and improvement of a robust smart card secured authentication scheme on SIP using elliptic curve cryptography, *Multimedia Tools Appl.*, **75** (2016), 4485–4504. doi: 10.1007/s11042-015-2487-7.
30. H. Arshad, M. Nikooghadam, Security analysis and improvement of two authentication and key agreement schemes for session initiation protocol, *J. Supercomput.*, **71** (2015), 3163–3180. doi: 10.1007/s11227-015-1434-8.
31. J. S. Tsai, Efficient nonce-based authentication scheme for session initiation protocol, *Int. J. Network Secur.*, **1** (2009), 12–16.
32. L. Zhang, S. Tang, S. Zhu, An energy efficient authenticated key agreement protocol for SIP-based green VoIP networks, *J. Network Comput. Appl.*, **59** (2016), 126–133. doi: 10.1016/j.jnca.2015.06.022.
33. AVISPA, *Automated validation of internet security protocols and applications*, Available from: http://www.avispa-project.org/.
34. *AVISPA web tool*, Available from: http://www.juniperresearch.com/viewpressrelease.php?pr=355.
35. N. Koblitz, A. Menezes, S. Vanstone, The state of elliptic curve cryptography, in *Designs Codes Cryptography*, **19** (2000), 173–193. doi: 10.1023/A:1008354106356.
36. M. Burrows, M. Abadi, R. M. Needham, A logic of authentication, *ACM Trans. Comput. Syst.*, **8** (1990), 18–36. doi: 10.1098/rspa.1989.0125.
37. Y. Lu, G. Xu, L. Li, Y. Yang, Robust privacy-preserving mutual authenticated key agreement scheme in roaming service for global mobility networks, *IEEE Syst. J.*, **13** (2019), 1454–1465. doi: 10.1109/JSYST.2018.2883349.
38. Y. Lu, M. Zhang, X. Zheng, An authentication framework in ICN-enabled industrial cyber-physical systems, in *International Conference on Security and Privacy in New Computing Environments*, (2021), 223–243. doi: 10.1007/978-3-030-66922-5_15.
39. PBC Library, *Pairing Based Cryptography*, Available from: http://crypto.stanford.edu/pbc/.
40. Ben Lynn, *On the Implementation of Pairing-Based Cryptography*, 2007. Available from: http://crypto.stanford.edu/pbc/thesis.pdf.
41. A. De Caro, V. Iovino, Java pairing based cryptography, in *Proceedings of the 16th IEEE Symposium on Computers and Communications*, (2011), 850–855.
42. M. Rao, T. Newe, I. Grout, A. Mathur, An FPGA-based reconfigurable IPSec AH core with efficient implementation of SHA-3 for high speed IoT applications, *Secur. Commun. Networks*, **9** (2016), 3282–3295. doi: 10.1002/sec.1533.
43. NIST, *National Institute of Standards and Technology (NIST)*, 2001. Available from: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

Appendix

```
role ui ( UI, S: agent,
          H,INV,MUL: hash_func,
          K: symmetric_key,
         Snd, Rcv: channel(dy)  )
played_by UI
def=    local
          State: nat,
          IDi,PWi,SK:text,
         X1,X2,X3:text,
          Ri1,Ri2,Ri3:text,
          R,Ri,N,AUTHs,AUTHu,P,Si :message
      const subs1,subs2,subs3 : protocol_id
      init   State := 0
      transition
    1. State = 0
    /\ Rcv(start)
    =|>
    State'   := 1
    /\ R' := new()
    /\ Snd({IDi}_K)
    /\ secret(IDi,subs1,{UI,S})
    /\ secret(PWi,subs2,{UI,S})
    2. State = 1
    /\ Rcv({MUL(INV(H(IDi,Si')),P').MUL(MUL(MUL(H(IDi),N'),Si'),P')}_K)
    =|>
    State'  := 2
    /\ Ri1' := new()
    /\ Snd({IDi.MUL(Ri1',P').MUL(Si',P')}_MUL(MUL(N',Si'),P'))
    3. State = 2
    /\ Rcv({MUL(Ri2',P').MUL(Ri1',P')}_H(IDi,Si').H(SK',MUL(Ri1',P'),Ri3').Ri3')
    =|>
    State'  := 3
    /\ AUTHu' := H(SK',MUL(Ri2',P'),Ri3')
    /\ Snd(AUTHu')
    /\ request(UI,S,si_ui_auths,H(SK',MUL(Ri1',P'),Ri3'))
    /\ witness(UI,S,si_ui_authu,AUTHu')
end role
```

Figure A1. The role of initiator.

```
role si ( UI,S : agent,
          H,INV,MUL  : hash_func,
          K  : symmetric_key,
          Snd, Rcv  : channel(dy)  )
played_by S
def=
          local
          State : nat,
          IDi,PWi,SK  :text,
          X1,X2,X3  :text,
          Ri1,Ri2,Ri3        :text,
          R,Ri,N,AUTHs,AUTHu,P,Si :message
          const subs1,subs2,subs3 : protocol_id
          init   State := 0
          transition
     1. State = 0
     /\ Rcv({IDi}_K)
     =|>
     State'    := 1
     /\ N' := new()
     /\ P' := new()
     /\ Si' := new()
     /\ X1' := MUL(INV(H(IDi,Si')),P')
     /\ X2' := MUL(MUL(MUL(H(IDi),N'),Si'),P')
     /\ Snd({X1'.X2'}_K)
     2. State = 1
     /\ Rcv({IDi.MUL(Ri1',P').MUL(Si',P')}_MUL(MUL(N',Si'),P'))
     =|>
     State'  := 2
     /\ Ri2'   := new()
     /\ Ri3'   := new()
     /\ SK'    := H(MUL(MUL(Ri2',Ri1'),P'),IDi)
     /\ AUTHs' := H(SK',MUL(Ri1',P'),Ri3')
     /\ Snd({MUL(Ri2',P').MUL(Ri1',P')}_H(IDi,Si').AUTHs'.Ri3')
     /\ secret(SK',subs3,{UI,S})
     /\ witness(S,UI,si_ui_auths,AUTHs')
      3. State = 2
     /\ Rcv(H(SK',MUL(Ri2',P'),Ri3'))
     =|>
     State'  := 3
     /\ request(S,UI,si_ui_authu,H(SK',MUL(Ri2',P'),Ri3'))
end role
```

Figure A2. The role of responder.

```
role session(UI,S : agent,
        H,INV,MUL  : hash_func,
         K : symmetric_key)
def=
  local
         S1,S2,R1,R2  : channel(dy)
  composition
        ui(UI, S, H, INV, MUL, K, S1, R1)
      /\ si(UI, S, H, INV, MUL, K, S2, R2)
end role
```

Figure A3. The role of session.

```
role environment()
 def=
  const
        si_ui_auths,si_ui_authu: protocol_id,
        ui, si : agent,
        h,int,mul: hash_func,
        k: symmetric_key
  intruder_knowledge = {ui, si, h, int, mul}
  composition
       session(ui, si, h, int, mul, k)
     /\session(ui, i , h, int, mul, k)
     /\session(i , si, h, int, mul, k)
end role
goal
  secrecy_of subs1
  secrecy_of subs2
  secrecy_of subs3
  authentication_on si_ui_auths
  authentication_on si_ui_authu
end goal
environment()
```

Figure A4. The role of environment and goal.