SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks 2016; **9**:3527–3542
Published online 7 July 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1558

RESEARCH ARTICLE

A novel and provably secure authentication and key agreement scheme with user anonymity for global mobility networks

Fan Wu[1]*, Lili Xu[2], Saru Kumari[3], Xiong Li[4], Ashok Kumar Das[5], Muhammad Khurram Khan[6], Marimuthu Karuppiah[7] and Renuka Baliyan[8]

[1] Department of Computer Science and Engineering, Xiamen Institute of Technology, Xiamen 361021, China
[2] School of Information Science and Technology, Xiamen University, Xiamen 361005, China
[3] Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar Pradesh 250004, India
[4] School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
[5] Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India
[6] Center of Excellence in Information Assurance, King Saud University, Saudi Arabia
[7] School of Computing Science and Engineering, VIT University, Vellore 632 014, India
[8] Department of Applied Science, JP Institute Of Engineering and Technology, Meerut, Uttar Pradesh 250001, India

ABSTRACT

Ubiquitous networks support the roaming service for mobile communication devices. The mobile user can use the services in the foreign network with the help of the home network. Mutual authentication plays an important role in the roaming services, and researchers put their interests on the authentication schemes. Recently, in 2016, Gope and Hwang found that mutual authentication scheme of He *et al.* for global mobility networks had security disadvantages such as vulnerability to forgery attacks, unfair key agreement, and destitution of user anonymity. Then, they presented an improved scheme. However, we find that the scheme cannot resist the off-line guessing attack and the de-synchronization attack. Also, it lacks strong forward security. Moreover, the session key is known to HA in that scheme. To get over the weaknesses, we propose a new two-factor authentication scheme for global mobility networks. We use formal proof with random oracle model, formal verification with the tool Proverif, and informal analysis to demonstrate the security of the proposed scheme. Compared with some very recent schemes, our scheme is more applicable. Copyright © 2016 John Wiley & Sons, Ltd.

KEYWORDS
mutual authentication; formal proof; global mobility network; smart card; formal verification

***Correspondence**
Fan Wu, Department of Computer Science and Engineering, Xiamen Institute of Technology, Xiamen 361021, China.
E-mail: conjurer1981@gmail.com

1. INTRODUCTION

Global roaming is now a widely used service for people who travel through different network regions. A mobile user (MU) can access various kinds of data from the foreign network with some mobile devices, such as PDAs and smartphones. Global mobility network (GLOMONET) makes this service applicable. Moreover, MU must register on its home network, or precisely the home agent (HA). However, as we all know, wireless communication channel is hazard. Mutual authentication between the entities involved in the session is one the most important points in security. When MU roams to other networks, he contacts the foreign network via the foreign agent (FA). FA then sends some critical data about MU and FA itself to HA to verify the identity of MU. HA checks them and sends the message to make FA and MU believe each other to be legal. At last, an interim session key is built as the symmetric encryption key applied in the latter part of conversation between FA and MU. Besides mutual authentication, how to keep the user's identity from being tracked is another critical issue. To protect the privacy of the user, identity should be kept away from the attacker's tracking. Researchers try various ways to hide the real identity in the public channel.

1.1. Related work

There are many authentication schemes for various sorts of communications, like key agreement schemes [1–14], and

signature schemes [15,16]. For wireless networks, especially in GLOMONETs, researchers have studied a lot about authentication schemes. Among them, two-factor authentication mechanism is a hot issue. Two-factor means that the user has a password and a smart card which is assigned by the registered server and stores data about the user's personal information. In 2004, Zhu and Ma [17] presented a two-factor authentication scheme providing user anonymity for wireless networks. Unluckily, in 2006, Lee *et al.* [18] showed that the scheme in [17] did not satisfy mutual authentication and was under the forgery attack, and they proposed an improved one. But in 2008, Wu *et al.* [19] pointed out that schemes in [17,18] both lacked user anonymity. In 2009, Chang *et al.* [20] considered that scheme in [18] could not resist the user impersonation attack and was devoid of user anonymity. Then, they presented a new one for GLOMONETs. Unfortunately, Youn *et al.* [21] criticized that the scheme in [20] was destitute of user anonymity in the same year. After that, He *et al.* [22] and Mun *et al.* [23] all considered that scheme in [19] was insecure, including weaknesses such as lacked of forward security and user anonymity, and vulnerability to the replay attack and the forgery attack. In 2013, Kim and Kawk [24] pointed out that Mun *et al.* scheme [23] could not withstand the replay attack and the man-in-the-middle attack. Still in the same year, Jiang *et al.* [25] and Shin *et al.* [26] presented their authentication schemes for GLOMONETs, respectively. However, Wen *et al.* [27] considered that the spoofing attack and the replay attack could be successfully carried out on the scheme in [25]. In 2015, Farash *et al.* [28] found that some security weaknesses such as lack of user anonymity and disclosure of the session key existed in [26,27]. Unluckily, we find that the improved scheme in [28] has disadvantages containing destitution of user anonymity and strong forward security, and susceptibility to off-line password guessing attack. Furthermore, Zhang *et al.* [29] showed that the hazard of tracking users could happen in [24], and they employed the elliptic curve cryptography (ECC) mechanism to build their own scheme. But we find that in Zhang *et al.* scheme, some of the session keys may be computed by the attacker, and the off-line guessing attack can be carried out. Still in 2015, Gope and Hwang [30] proposed a new two-factor authentication scheme for GLOMONETs. Unfortunately, we have found that the scheme in [30] has weaknesses containing susceptibility to the de-synchronization attack, session key known to HA, and impracticality due to normal network time delay.

1.2. Motivations and contributions

In 2016, Gope and Hwang [31] showed that the disadvantages including unfair session key agreement, track of the user and impracticality, and susceptibility to forgery attack occur in the scheme of [22]. Then, an enhance version for GLOMONETs was presented. In fact, it still has weaknesses in the security aspect. To overcome the aforementioned all security problems, a really secure mutual authentication scheme providing user anonymity should be proposed, and it should be analyzed to be true in different perspectives.

The contributions of this paper are as follows:

(1) We show that the scheme in [31] has security disadvantages containing vulnerability to the off-line guessing attack and the de-synchronization attack, session key disclosure to HA, and lack of strong forward security.
(2) Like the scheme in [29], a new mutual authentication scheme for GLOMONETs also based on ECC is proposed.
(3) Via the proofs with random oracle model, the cryptographic tool Proverif, and the informal analysis, our scheme is secure and applicable.

1.3. Organization of the paper

The rest of the paper is arranged as follows: Section 2 introduces the basic knowledge for this paper. Sections 3 and 4 are the scheme in [31] and its weaknesses, respectively. Sections 5, 6, 7 and 8 illustrate our scheme with its formal proof, formal verification, and informal security explanation, respectively. We show the performance comparison among our scheme and some recent schemes in Section 9. Finally, the conclusion appears in Section 10.

2. PRELIMINARIES

2.1. Terminology

The notations used throughout the paper are presented in Table I. The symbols in the last three rows are for parts of scheme in [31] and ours.

2.2. Basic knowledge for elliptic curve cryptography

The elliptic curve E is as follows: $y^2 = x^3 + ax + b \ mod \ p$ where $a, b \in F_p$ and $4a^3 + 27b^2 \ mod \ p \neq 0$. Here, "*mod p*" is omitted in the rest part. Some necessary basic problems are demonstrated as follows:

- Elliptic curve discrete logarithm problem: If R and Q are two points in the group G, it is difficult to calculate the special integer $\alpha \in Z_n^*$ such that $Q = \alpha R$.
- Elliptic curve Diffie–Hellman (ECDH) problem: If $\alpha, \beta \in Z_n^*$ and αP and βP are points in G, it is difficult to compute $\alpha\beta P$ with only αP and βP. The symbol $Adv_{\mathcal{A}}^{ECDH}(t)$ means that the probability for $\mathcal{A}$ working out the ECDH problem with polynomial time t.

2.3. Assumptions for informal analysis

Assumption 1. *Based on* [32], *the passwords of users are in a finite set, and so do the identities.* $\mathcal{A}$ *can guess them*

DOI: 10.1002/sec

Table I. Notations.

Symbol	Meaning
ID_{MU}, PW_{MU}	The mobile user's identity and password
ID_{FA}	The foreign agent's identity
ID_{HA}, x	The home agent's identity and its secret key
K_{fh}	The secret key owned by FA and HA in common
$\mathcal{A}$	The adversary
$E_k(.)/D_k(.)$	The symmetric encryption/decryption function with key k
$h(.)$	The one-way hash function
p	A large prime
$E(F_p)$	An elliptic curve E with a finite field F_p on it
G, P, n	An additive subgroup on F_p with its generator and prime order
l	The security length
SK_{FA}, SK_{MU}	Session keys constructed by FA and MU, respectively
$\|\|$	The concatenation operation
$\oplus$	The exclusive or operation
$\xrightarrow{M}, \xleftarrow{M}$	The message M transmitted via a public channel
$\Longrightarrow^{M}, \Longleftarrow^{M}$	The message M transmitted via a secure channel
$Data^{old} \leftarrowtail Data^{new}$	Data update, from $Data^{old}$ to $Data^{new}$

simultaneously in polynomial time. But the secret keys, like K_{fh} and x, the hash results and the random numbers are difficult to guess. Moreover, it is difficult to obtain the collision of hash results within polynomial time.

Assumption 2. *According to* [10,12,33–35]*, we consider that the data in MU's smart card can be obtained once and analyzed by $\mathcal{A}$.*

Assumption 3. *According to* [10,12]*, to keep the two-factor authentication environment, until the data in the smart card are obtained, $\mathcal{A}$ can control the public channel, or the insecure channel, with activities such as eavesdropping, intercepting, modifying, and forging messages in it. In many past papers, such as* [1,26,32]*, there is no limitation about the attacker in the model. However, such premise only fits for the formal proof, which is with random oracle model, for example, Section 6 of this paper. The aim of the formal proof is to calculate the advantage of the attacker $\mathcal{A}$ judging if the output string in the final Test query is the session key, but it is not fit for the informal analysis. Roughly speaking, the aim of the attacker $\mathcal{A}$ in the informal security analysis is to break both factors in two-factor authentication environment. If an adversary $\mathcal{A}$ retrieves data from MU's smart card at one time and then still goes on eavesdropping MU's session, it means that the security responsibility which should lie on both factors could only depend on the remaining factor password. In fact, the usual demonstration "the security of two-factor authentication environment should be kept even if one factor is compromised" is ambiguous. The accurate criteria of two-factor authentication security should be illustrated as follows: if preparations including enough information from the public channel and one compromised factor after message collection are ready for the attacker $\mathcal{A}$ while the second factor still cannot be cracked under such conditions, the two-factor authentication environment is secure. Certainly, $\mathcal{A}$ cannot do anything if he only obtains the password, so we employ the model as we show at the start of this assumption. Furthermore, $\mathcal{A}$ cannot obtain any message from the secure channel, which is used in the registration phase.*

Assumption 4. *$\mathcal{A}$ can obtain the past session keys. For this assumption, we focus on the relation among the session keys.*

Assumption 5. *Based on* [6]*, to guarantee the property of strong forward security, even if all the long-term data of the entities, such as MU's identity and password, and secret keys of FA and HA, are leaked to $\mathcal{A}$, $\mathcal{A}$ cannot calculate the past session keys. When this item is considered, Assumption 4 does not occur. For this assumption, the ability of calculating the past session keys is considered.*

3. REVIEW OF SCHEME IN [31]

There are three phases in Gope and Hwang's scheme: registration, mutual authentication and key agreement (MAKA) and password renewal. At first, HA and FA share a common private key K_{fh}.

3.1. Registration

- Step 1: MU selects his identity ID_{MU} and sends it to HA via a secure manner.
- Step 2: After receiving ID_{MU}, HA computes the string $B_0 = h(ID_{MU}||x) \oplus ID_{HA}$ and generates a pseudo-id set $PID = \{PID_1, PID_2 \ldots\}$. Each $PID_j = h(ID_{MU}||r_j||B_0)$ and r_j is a random number. HA also generates a 64-bit string Tr_{seq} as a sequence number. Finally, HA stores PID, B_0, ID_{MU}, and Tr_{seq} in its database, injects Tr_{seq}, PID, ID_{HA}, and B_0 into

a smart card, and sends the smart card to MU secretly. Here, if the case that the sequence number in HA is different from the number sent by MU happens, the session should be denied, and the synchronization operation should be carried out.

- Step 3: MU selects a password PW_{MU} and computes the data $B_1 = B_0 \oplus h(ID_{MU}||PW_{MU})$ and $\overline{PID} = PID \oplus h(ID_{MU}||PW_{MU})$. Finally, MU replaces (B_0, PID) with $(B_1, \overline{PID})$.

3.2. Mutual authentication and key agreement

We list the MAKA phase with five steps and show them in Table II. Because the messages are transmitted in an insecure way, we use

- Step 1: MU inputs ID_{MU} and PW_{MU}. The smart card generates two random numbers N_{m1} and N_{m2}, and computes $C_1 = N_{m1} \oplus N_{m2}$, $B_0 = B_1 \oplus h(ID_{MU}||PW_{MU})$, $C_2 = E_{B_0}(N_{m2}||C_1)$ and $C_3 = h(ID_{MU}||B_0||N_{m1}||Tr_{seq})$. Then, the message $M_1 = \{Tr_{seq}, ID_{HA}, C_2, C_3\}$ is sent to FA. However, if it is the chance of synchronization, the smart card picks one unused $\overline{PID_j}$, computes $PID_j = \overline{PID_j} \oplus h(ID_{MU}||PW_{MU})$ and C_2 as previously, makes $C_3 = PID_j$, and sends the message $M_1 = \{Tr_{seq}, ID_{HA}, C_2, C_3\}$ to FA.
- Step 2: When receiving M_1, FA generates two random numbers N_{f1} and N_{f2} and computes $C_4 = N_{f1} \oplus N_{f2}$, $C_5 = E_{K_{fh}}(N_{f2}||C_4)$ and $C_6 = h(M_1||K_{fh}||C_4)$. After that, FA sends the message $M_2 = \{Tr_{seq}, C_2, C_3, C_5, C_6, ID_{FA}\}$ to HA.
- Step 3: After receiving M_2, and if Tr_{seq} exists, HA will check it. If it can be found, HA obtains (B_0, ID_{MU}) according to Tr_{seq} in its database. HA decrypts (C_2, C_5) with (B_0, K_{fh}), respectively. After verifying C_3 and C_6, HA generates a new sequence number Tr_{seq}^{new} and then computes the following data: $C_7 = E_{K_{fh}}(N_{m1}||N_{f1})$, $C_8 =$

Table II. Mutual authentication and key agreement phase in [31].

Mobile user	Foreign agent	Home agent
input ID_{MU}, PW_{MU} generates N_{m1}, N_{m2} $C_1 = N_{m1} \oplus N_{m2}$ $B_0 = B_1 \oplus h(ID_{MU}\|\|PW_{MU})$ $C_2 = E_{B_0}(N_{m2}\|\|C_1)$ $C_3 = h(ID_{MU}\|\|B_0\|\|N_{m1}\|\|Tr_{seq})$ $\xrightarrow{M_1=\{Tr_{seq},ID_{HA},C_2,C_3\}}$		
	generate N_{f1}, N_{f2} $C_4 = N_{f1} \oplus N_{f2}$ $C_5 = E_{K_{fh}}(N_{f2}\|\|C_4)$ $C_6 = h(M_1\|\|K_{fh}\|\|C_4)$ $\xrightarrow{M_2=\{Tr_{seq},C_2,C_3,C_5,C_6,ID_{FA}\}}$	
		if Tr_{seq} appears, get (B_0, ID_{MU}) else search PID_j and get (B_0, ID_{MU}) decrypt (C_2, C_5) with (B_0, K_{fh}) verify C_3 and C_6 generate Tr_{seq}^{new} $C_7 = E_{K_{fh}}(N_{m1}\|\|N_{f1})$ $C_8 = h(C_7\|\|K_{fh}\|\|N_{f1})$ $C_9 = h(B_0\|\|ID_{MU}\|\|N_{m1}) \oplus Tr_{seq}^{new}$ $C_{10} = h(N_{m1}\|\|B_0\|\|ID_{MU}) \oplus N_{f1}$ $C_{11} = h(C_{10}\|\|N_{m2}\|\|B_0\|\|C_9)$ $Tr_{seq} \leftarrow\!\prec Tr_{seq}^{new}$ $\xleftarrow{M_3=\{C_7,C_8,C_9,C_{10},C_{11}\}}$
	decrypt C_7 verify $C_8? = h(C_7\|\|K_{fh}\|\|N_{f1})$ $SK_{FA} = N_{m1} \oplus N_{f1}$ $\xleftarrow{M_3=\{C_9,C_{10},C_{11}\}}$	
check $C_{11}? = h(C_{10}\|\|N_{m2}\|\|B_0\|\|C_9)$ $N_{f1} = C_{10} \oplus h(N_{m1}\|\|B_0\|\|ID_{MU})$ $Tr_{seq}^{new} = C_9 \oplus h(B_0\|\|ID_{MU}\|\|N_{m1})$ $SK_{MU} = N_{m1} \oplus N_{f1}$ $Tr_{seq} \leftarrow\!\prec Tr_{seq}^{new}$		

DOI: 10.1002/sec

$h(C_7||K_{fh}||N_{f1})$, $C_9 = h(B_0||ID_{MU}||N_{m1}) \oplus Tr_{seq}^{new}$, $C_{10} = h(N_{m1}||B_0||ID_{MU}) \oplus N_{f1}$ and $C_{11} = h(C_{10}||N_{m2}||B_0||C_9)$. Here, HA must update Tr_{seq} with Tr_{seq}^{new}. The paper [31] does not mention that, and we add it here. Then, $M_3 = \{C_7, C_8, C_9, C_{10}, C_{11}\}$ is sent to FA. Otherwise, if Tr_{seq} does not appear, HA searches PID_j in database to find (B_0, ID_{MU}) and goes on the aforementioned operations.

- Step 4: After receiving M_3, FA decrypts C_7 and verifies $C_8? = h\left(C_7||K_{fh}||N_{f1}\right)$. If it is true, HA computes $SK_{FA} = N_{m1} \oplus N_{f1}$ and sends $M_3 = \{C_9, C_{10}, C_{11}\}$ to MU.
- Step 5: When receiving M_4, MU checks $C_{11}? = h(C_{10}||N_{m2}||B_0||C_9)$. If it is right, the following data $N_{f1} = C_{10} \oplus h(N_{m1}||B_0||ID_{MU})$, $Tr_{seq}^{new} = C_9 \oplus h(B_0||ID_{MU}||N_{m1})$, and the session key $SK_{MU} = N_{m1} \oplus N_{f1}$ are calculated, and the detail that MU updates Tr_{seq} with Tr_{seq}^{new} at last should be replenished because [31] does not mention it.

3.3. Password renewal

When MU wants to change his password, he inputs his identity ID_{MU}, password PW_{MU}, and a new password PW_{MU}^{new}. Then, the smart card computes the data $B_1^{new} = B_1 \oplus h(ID_{MU}||PW_{MU}) \oplus h\left(ID_{MU}||PW_{MU}^{new}\right)$ and $\overline{PID}^{new} = \overline{PID} \oplus h(ID_{MU}||PW_{MU}) \oplus h\left(ID_{MU}||PW_{MU}^{new}\right)$ and updates $(B_1, \overline{PID})$ with $\left(B_1^{new}, \overline{PID}^{new}\right)$.

4. WEAKNESSES OF SCHEME IN [31]

4.1. Off-line guessing attack

Suppose $\mathcal{A}$ guesses $\left(ID^*, PW^*\right)$, eavesdrops $\left\{M_1^{old}, M_2^{old}, M_3^{old}, M_4^{old}\right\}$ from the public channel and then retrieves $(\overline{PID}, B_1, Tr_{seq})$ from MU's smart card at once. He can use the following equations: $B_0^* = B_1 \oplus h(ID^*||PW^*)$ and $\left(N_{m2}^*||C_1^*\right) = D_{B_0^*}\left(C_2^{old}\right)$. Then, HA can check if $C_3^{old} = h\left(ID^*||B_0^*||\left(C_1^* \oplus N_{m2}^*\right)||Tr_{seq}^{old}\right)$ or $Tr_{seq} = C_9^{old} \oplus h\left(B_0^*||ID^*||\left(C_1^* \oplus N_{m2}^*\right)\right)$ until the pair $\left(ID^*, PW^*\right)$ is correct.

4.2. De-synchronization attack

De-synchronization attack means that a legal user in the system cannot login to the remote server normally because the data stored on his side are different to the relative data stored in the server, or some inconsistency happens in different entities involved in a session. In the password renewal phase, it is obvious that the validity of old password is not checked. If MU inputs a wrong password PW^{wrong} by accident, the smart card will compute $B_1^{wrong} = B_1^{old} \oplus h(ID_{MU}||PW^{wrong}) \oplus h\left(ID_{MU}||PW_{MU}^{new}\right)$, and $PID^{wrong} = \overline{PID}^{old} \oplus h\left(ID_{MU}||PW^{wrong}\right) \oplus h\left(ID_{MU}||PW_{MU}^{new}\right)$. Note that B_0 is stored in HA. In the next session, MU can only obtain $B_0^{wrong} = B_1^{wrong} \oplus h\left(ID_{MU}||PW_{MU}^{new}\right) \neq B_0$, and HA cannot obtain the correct C_1 and N_{m2} by decrypting C_2, not to say N_{m1}, which equals $C_1 \oplus N_{m2}$ and is required to build the session key on the FA side, so the fault may lead to the block of legal roaming access.

4.3. Session key known to home agent

This property is applied first in [29]. The session key is constructed by $N_{m1} \oplus N_{f1}$, and the two strings N_{m1} and N_{f1} can be calculated by HA because both of them need protecting by the secret key x in the public channel. Thus, a malicious administrator of HA has the ability to compute every session key.

4.4. Lack of strong forward security

Like the analysis in Section 4.3, if the secret key x in HA is obtained by $\mathcal{A}$, N_{m1}, and N_{f1} in any session can be obtained, too, the past session keys could be calculated.

Table III. Registration of the proposed scheme.

Mobile user	Home agent
select ID_{MU}, PW_{MU}, b_{MU}	
$HPW_{MU} = h(PW_{MU}||b_{MU})$	
$\xrightarrow{(ID_{MU}, HPW_{MU})}$	
	generate PID_{MU}
	store ID_{MU}
	$B_1 = h(PID_{MU}||ID_{HA}||x) \oplus HPW_{MU}$
	$B_2 = h(ID_{MU}||x) \oplus h(ID_{MU}||HPW_{MU})$
	smart card $\Leftarrow (B_1, B_2, PID_{MU}, ID_{HA})$
	$\xleftarrow{\textit{smart card}}$
$B_3 = h(ID_{MU}||PW_{MU}) \oplus b_{MU}$	
smart card $\Leftarrow B_3$	

5. PROPOSED SCHEME

Our scheme is also divided into three phases as the same as Gope and Hwang's in [31]. At first, HA and FA share a common private key K_{fh}.

5.1. Registration

We show this phase in Table III.

- Step 1: MU selects his identity ID_{MU} and password PW_{MU}, generates a random number b_{MU}, computes $HPW_{MU} = h(PW_{MU}||b_{MU})$, and sends (ID_{MU}, HPW_{MU}) to HA via a secure manner.
- Step 2: HA generates a random string PID_{MU} as a pseudo-identity, stores ID_{MU} into its database, computes the following data $B_1 = h(PID_{MU}||ID_{HA}||x) \oplus HPW_{MU}$ and $B_2 = h(ID_{MU}||x) \oplus h(ID_{MU}||HPW_{MU})$, and injects $(B_1, B_2, PID_{MU}, ID_{HA})$ into a smart card. Finally, HA sends the smart card to MU through a secure manner.
- Step 3: When MU receives the smart card, he computes $B_3 = h(ID_{MU}||PW_{MU}) \oplus b_{MU}$ and stores B_3 in the smart card.

Table IV. Mutual authentication and key agreement phase of the proposed scheme.

Mobile user	Foreign agent	Home agent
input ID_{MU} and PW_{MU} generate $\alpha \in Z_n^*$, N_m $b_{MU} = B_3 \oplus h(ID_{MU}\|\|PW_{MU})$ $HPW_{MU} = h(PW_{MU}\|\|b_{MU})$ $C_1 = B_1 \oplus HPW_{MU} \oplus N_m$ $C_2 = h(N_m\|\|PID_{MU}\|\|ID_{HA}) \oplus ID_{MU}$ $C_3 = \alpha P$ $C_4 = h(N_m\|\|PID_{MU}\|\|ID_{MU}\|\|C_3)$ $\xrightarrow{M_1=\{PID_{MU},C_1,C_2,C_3,C_4,ID_{HA}\}}$		
	generate $\beta \in Z_n^*$ and N_f $C_5 = \beta P$ $C_6 = E_{K_{fh}}(N_f) \oplus h(C_3\|\|C_5)$ $C_7 = h(PID_{MU}\|\|C_1\|\|C_2\|\|C_3\|\|C_4\|\|C_5\|\|N_f)$ $\xrightarrow{M_2=\{PID_{MU},C_1,C_2,C_3,C_4,C_5,C_6,C_7,ID_{FA}\}}$	
		$N_m = C_1 \oplus h(PID_{MU}\|\|ID_{HA}\|\|x)$ $ID_{MU} = C_2 \oplus h(N_m\|\|PID_{MU}\|\|ID_{HA})$ search ID_{MU} check $C_4? = h(N_m\|\|PID_{MU}\|\|ID_{MU}\|\|C_3)$ $N_f = D_{K_{fh}}(C_6 \oplus h(C_3\|\|C_5))$ check $C_7? = h(PID_{MU}\|\|C_1\|\|C_2\|\|C_3\|\|C_4\|\|C_5\|\|N_f)$ generate PID_{MU}^{new} $C_8 = h(ID_{MU}\|\|x)$ $C_9 = h(PID_{MU}^{new}\|\|ID_{HA}\|\|x)$ $C_{10} = C_9 \oplus h(C_8\|\|N_m)$ $C_{11} = h(C_8\|\|C_3\|\|C_5\|\|N_m) \oplus PID_{MU}^{new}$ $C_{12} = h(C_9\|\|N_m\|\|ID_{MU}\|\|PID_{MU}\|\|PID_{MU}^{new}\|\|ID_{HA})$ $C_{13} = h(K_{fh}\|\|N_f\|\|C_3\|\|C_5\|\|ID_{FA}\|\|ID_{HA})$ $\xleftarrow{M_3=\{C_{10},C_{11},C_{12},C_{13}\}}$
	check $C_{13}? = h(K_{fh}\|\|N_f\|\|C_3\|\|C_5\|\|ID_{FA}\|\|ID_{HA})$ $SK_{FA} = h(C_3\|\|C_5\|\|\beta C_3)$ $C_{14} = h(SK_{FA}\|\|PID_{MU}\|\|ID_{HA})$ $\xleftarrow{M_4=\{C_5,C_{10},C_{11},C_{12},C_{14}\}}$	
$C_{15} = B_2 \oplus h(ID_{MU}\|\|HPW_{MU})$ $C_{16} = C_{10} \oplus h(C_{15}\|\|N_m)$ $PID_{MU}^{new} = C_{11} \oplus h(C_{15}\|\|C_3\|\|C_5\|\|N_m)$ check $C_{12}? = h(C_{16}\|\|N_m\|\|ID_{MU}\|\|PID_{MU}\|\|PID_{MU}^{new}\|\|ID_{HA})$ $SK_{MU} = h(C_3\|\|C_5\|\|\alpha C_5)$ check $C_{14}? = h(SK_{MU}\|\|PID_{MU}\|\|ID_{HA})$ $B_1^{new} = C_{16} \oplus HPW_{MU}$ $(B_1, PID_{MU}) \longleftarrow (B_1^{new}, PID_{MU}^{new})$		

DOI: 10.1002/sec

5.2. Mutual authentication and key agreement

The MAKA phase is illustrated in Table IV.

- Step 1: MU inputs ID_{MU} and PW_{MU}. The smart card generates random numbers $\alpha \in Z_n^*$ and N_m. Then, the following data are calculated: $b_{MU} = B_3 \oplus h(ID_{MU}||PW_{MU})$, $HPW_{MU} = h(PW_{MU}||b_{MU})$, $C_1 = B_1 \oplus HPW_{MU} \oplus N_m$, $C_2 = h(N_m||PID_{MU}||ID_{HA}) \oplus ID_{MU}$, $C_3 = \alpha P$, and $C_4 = h(N_m||PID_{MU}||ID_{MU}||C_3)$. After the aforementioned calculations, MU sends the message $M_1 = \{PID_{MU}, C_1, C_2, C_3, C_4, ID_{HA}\}$ to FA.
- Step 2: FA generates random numbers $\beta \in Z_n^*$ and N_f, and calculates the data as follows: $C_5 = \beta P$, $C_6 = E_{K_{fh}}(N_f) \oplus h(C_3||C_5)$ and $C_7 = h(PID_{MU}||C_1||C_2||C_3||C_4||C_5||N_f)$. After the aforementioned calculations, FA sends the message $M_2 = \{PID_{MU}, C_1, C_2, C_3, C_4, C_5, C_6, C_7, ID_{FA}\}$ to HA.
- Step 3: After receiving M_2, HA computes $N_m = C_1 \oplus h(PID_{MU}||ID_{HA}||x)$ and $ID_{MU} = C_2 \oplus h(N_m||PID_{MU}||ID_{HA})$. Then, it searches ID_{MU} in the database for auditing and checks $C_4? = h(N_m||PID_{MU}||ID_{MU}||C_3)$. If it is right, HA then executes the decryption to calculate the string $N_f = D_{K_{fh}}(C_6 \oplus h(C_3||C_5))$ and checks $C_7? = h(PID_{MU}||C_1||C_2||C_3||C_4||C_5||N_f)$. If so, HA generates PID_{MU}^{new} as a new pseudo-identity and calculates $C_8 = h(ID_{MU}||x)$, $C_9 = h(PID_{MU}^{new}||ID_{HA}||x)$, $C_{10} = C_9 \oplus h(C_8||N_m)$, $C_{11} = h(C_8||C_3||C_5||N_m) \oplus PID_{MU}^{new}$, $C_{12} = h(C_9||N_m||ID_{MU}||PID_{MU}||PID_{MU}^{new}||ID_{HA})$ and $C_{13} = h(K_{fh}||N_f||C_3||C_5||ID_{FA}||ID_{HA})$. Finally, the message $M_3 = \{C_{10}, C_{11}, C_{12}, C_{13}\}$ is sent to FA.
- Step 4: After receiving M_3, FA checks $C_{13}? = h(K_{fh}||N_f||C_3||C_5||ID_{FA}||ID_{HA})$. If it is right, FA computes $SK_{FA} = h(C_3||C_5||\beta C_3)$ and $C_{14} = h(SK_{FA}||PID_{MU}||ID_{HA})$ and sends the message $M_4 = \{C_5, C_{10}, C_{11}, C_{12}, C_{14}\}$ to MU.
- Step 5: After MU receives M_4, the smart card computes $C_{15} = B_2 \oplus h(ID_{MU}||HPW_{MU})$, $C_{16} = C_{10} \oplus h(C_{15}||N_m)$ and $PID_{MU}^{new} = C_{11} \oplus h(C_{15}||C_3||C_5||N_m)$ and checks $C_{12}? = h\left(C_{16}||N_m||ID_{MU}||PID_{MU}||PID_{MU}^{new}||ID_{HA}\right)$.
  If it is true, the smart card computes $SK_{MU} = h(C_3||C_5||\alpha C_5)$ and checks $C_{14}? = h(SK_{MU}||PID_{MU}||ID_{HA})$. If it is right, the smart card computes $B_1^{new} = C_{16} \oplus HPW_{MU}$ and updates (B_1, PID_{MU}) with $\left(B_1^{new}, PID_{MU}^{new}\right)$ at last.

5.3. Password renewal

- Step 1: MU inputs ID_{MU} and PW_{MU}, calculates b_{MU} and HPW_{MU} as in MAKA phase. Then, the random number N_m is produced, and the following data C_1, C_2, C_{15}, and $C_{17} = h(N_m||PID_{MU}||ID_{MU}||C_{15})$ are calculated. At last, MU sends $M_5 = \{PID_{MU}, C_1, C_2, C_{17}\}$ with a password change request to HA.
- Step 2: After M_5 arrives, HA calculates N_m and ID_{MU} and checks if ID_{MU} is in its database. If it exists, C_7 is calculated and $C_{17}? = h(N_m||PID_{MU}||ID_{MU}||C_7)$ is verified. If the equation is correct, HA generates a new random string PID_{MU}^{new} and computes the following data C_9, $C_{18} = h(N_m||ID_{MU}||PID_{MU}||C_7) \oplus PID_{MU}^{new}$, $C_{19} = h(C_7||N_m||PID_{MU}^{new}||ID_{MU}) \oplus C_9$ and $C_{20} = h(C_7||N_m||PID_{MU}^{new}||C_9||ID_{MU})$. Finally, a message $M_6 = \{C_{18}, C_{19}, C_{20}\}$ with a permission grant is sent to MU.
- Step 3: The smart card computes $PID_{MU}^{new} = C_{18} \oplus h(N_m||ID_{MU}||PID_{MU}||C_{15})$ and $C_{21} = C_{19} \oplus h(C_{15}||N_m||PID_{MU}^{new}||ID_{MU})$ and checks $C_{20}? = h(C_{15}||N_m||PID_{MU}^{new}||C_{21}||ID_{MU})$. If it is right, MU inputs a new password PW_{MU}^{new}, and the smart card generates a new nonce b_{MU}^{new} and calculates $HPW_{MU}^{new} = h(PW_{MU}^{new}||b_{MU}^{new})$, $B_1^{new2} = C_{21} \oplus HPW_{MU}^{new}$, $B_2^{new} = C_{14} \oplus h\left(ID||HPW_{MU}^{new}\right)$, and $B_3^{new} = h\left(ID_{MU}||PW_{MU}^{new}\right) \oplus b_{MU}^{new}$. Finally, the smart card replaces $(PID_{MU}, B_1, B_2, B_3)$ with $\left(PID_{MU}^{new}, B_1^{new2}, B_2^{new}, B_3^{new}\right)$ in its memory.

6. FORMAL SECURITY ANALYSIS OF OUR SCHEME

6.1. Basis of the model

Based on [6,10,36], we illustrate the model of the proof in the succeeding text.

To make the attacker $\mathcal{A}$ convenient and the process simple, we suppose only three participants exist in the protocol $\mathcal{P}$, containing an MU, an HA, and an FA, so we consider ID_{MU}, ID_{HA}, ID_{FA}, P, and n are public in the proof. Moreover, the three entities have their own secret information: MU has his password PW_{MU} and smart card, HA has the secret keys x and K_{fh}, and FA has K_{fh}. $\mathcal{I}$ may denote any entity if we need not differentiate it with others. Every instance for $\mathcal{I}$ has one number of its own, for example, we use MU^i to express the *i–th* instance of MU. Likewise, FA^j, HA^t, and $\mathcal{I}^k$ could be comprehended. Each instance could be seen as an oracle, and three results may happen: *accept*, *reject*, and $\perp$. *accept* occurs after an oracle receiving a correct message; *reject* occurs after the wrong message arrives; $\perp$ occurs when the oracle does not output any answer. Once the *accept* appears in MU^i or FA^j and a session key is built, the following data are formed: the session identity sid_{MU^i} or sid_{FA^j}, the partner pid_{MU^i} or pid_{FA^j}, and the session key SK_{MU^i} or SK_{FA^j}. If MU^i and FA^j both build the same session key, we define either of them has the state *Partnering*, and they are each other's partners. The relations between the partners are as follows: $sid_{MU^i} = sid_{FA^j}$,

$pid_{MU^i} = FA^j$, $pid_{FA^j} = MU^i$, and $SK_{MU^i} = SK_{FA^j}$. We also consider SK is a session key, which we do not care about the source.

To crack the session key or the MAKA process, $\mathcal{A}$ can use a simulator to ask for the following queries:

- *Send*$(\mathcal{I}, \mathcal{I}_r^i, m)$: The oracle $\mathcal{I}_r^i$ receives the message m from the entity $\mathcal{I}$. If m is a correct message and $\mathcal{I}_r^i$ is ready to receive, the simulator will response as in $\mathcal{P}$. Otherwise, the query will be abandoned.
- *Execute*(MU^i, FA^j, HA^t): It denotes the MAKA process. All messages among MU^i, FA^j, and HA^t are eavesdropped by $\mathcal{A}$.
- *Reveal*$(\mathcal{I}^k)$: If MU^i or FA^j has a session key, $\mathcal{A}$ can gain it by this query.
- *Corrupt*$(MU^i, smart\ card)$: $\mathcal{A}$ can use the query to simulate the accident of the smart card loss.
- *Corrupt*$(\mathcal{I}^k)$: It is for the strong forward security [6] and simulates the accident that all the long-term data are lost for $\mathcal{I}^k$.
- *Test*$(\mathcal{I}^k)$: Finally, $\mathcal{A}$ must select a session to challenge. $\mathcal{I}$ stands for MU or FA. If $\mathcal{I}^k$ does not come to the status of *accept* or is not suitable for *sfs – fresh* explained later, $\perp$ will be the result. Otherwise, a bit b is chosen. If $b = 1$, SK will be the answer; otherwise, a random string $\{0,1\}^l$ will be output.

Furthermore, some definitions are demonstrated based on [6]:

- *sfs – fresh*(**S**trong **F**orward **S**ecurity-fresh): It is for MU and FA. $\mathcal{I}^k$ will not be *sfs – fresh* if any of the accidents happens:
  - A *Reveal*$(\mathcal{I}^k)$ or *Reveal*$(pid_{\mathcal{I}^k})$ happens.
  - *Corrupt*$(\mathcal{I}^k)$ or *Corrupt*$(pid_{\mathcal{I}^k})$ occurs before *Test*$(\mathcal{I}^k)$.
- *sfs – secure*(**S**trong **F**orward **S**ecurity-secure): The advantage of breaking the session key of $\mathcal{P}$ for the adversary $\mathcal{A}$ is the probability of correctly guessing the bit b after *Test*$(\mathcal{I}^k)$. $Adv_{\mathcal{P}}^{sfs}(\mathcal{A}) = 2|Pr[b = b'] - \frac{1}{2}|$ denotes the probability where b' is the bit guessed by $\mathcal{A}$. The quantity of passwords is $|\mathcal{D}|$, and q_s is the number of *Send* queries. If $Adv_{\mathcal{P}}^{sfs}(\mathcal{A})$ is slightly larger than $\frac{O(q_s)}{|\mathcal{D}|}$ with the security length l, $\mathcal{P}$ is *sfs – secure*.

6.2. Process of the proof

Theorem 1. *An additive group G with order n on $E(F_p)$ is employed. Let the number of passwords be $|\mathcal{D}|$ and l be the shortest length for random numbers and hash results. The adversary $\mathcal{A}$ has chances including q_s Send queries, q_e Execute queries and q_h hash queries to break the security of $\mathcal{P}$ in polynomial time t as the upper-bound. Then, the advantage of cracking the session key of the sfs – secure protocol $\mathcal{P}$ is*

$$Adv_{\mathcal{P}}^{sfs}(\mathcal{A}) \leq \frac{(q_s+q_e)^2}{n-1} + \frac{(q_s+q_e)^2+q_h^2}{2^l} + \frac{7q_s+15q_h}{2^{l-1}} + \frac{2q_s}{|\mathcal{D}|} + 4q_h\left((q_s+q_e)^2+1\right)Adv_{\mathcal{A}}^{ECDH}(t')$$

where T_m is the time for one point multiplication in G and $t' = t + (2q_s + 4q_e)T_m$.

Proof 1. *There are six games from $\mathcal{G}_0$ to $\mathcal{G}_5$ in sequence to describe the proof. $Succ_i$ means the probability for $\mathcal{A}$ correctly guessing b in game $\mathcal{G}_i$. According to Section 6.1, the identity of MU needs not guessing because only one MU exists and ID_{MU} is made public.*

- *Game $\mathcal{G}_0$: This game simulates the real scheme under random oracles environment. We can see $Adv_{\mathcal{P}}^{sfs}(\mathcal{A}) = 2|Pr[Succ_0] - \frac{1}{2}|$. Here, b' is a chosen bit when $\mathcal{A}$ uses more queries or time than upper-bound or the games is over without $\mathcal{A}$'s answer.*
- *Game $\mathcal{G}_1$: It simulates all queries mentioned in Section 6.1, but we should say that Send queries consist of five cases: Send$(init, MU^i, \Phi)$, Send $\left(MU^i, FA^j, M_1\right)$, Send$(FA^j, HA^t, M_2)$, Send$(HA^t, FA^j, M_3)$, and Send$(FA^j, MU^i, M_4)$. The corresponding relationships are just between the five Send queries and the steps in MAKA phase. Moreover, three lists are needed: $\mathcal{L}_h$ stores the pair of queried string and hash value; $\mathcal{L}_{\mathcal{A}}$ stores the pair of the hash query string and result asked by $\mathcal{A}$; $\mathcal{L}_{\mathcal{P}}$ stores transcripts in the simulations. Here, the work process of hash oracle should be introduced: if m is a string and h(m) is asked, the simulator searches the record (m, r) first. If it is in $\mathcal{L}_h$, r is the result. Otherwise, a random string $r \in \{0,1\}^l$ is produced and returned. Also, (m, r) is written in $\mathcal{L}_h$, so $\mathcal{A}$ cannot tell apart $\mathcal{G}_1$ from $\mathcal{G}_0$, and $Pr[Succ_1] = Pr[Succ_0]$.*
- *Game $\mathcal{G}_2$: It contains the cases of collisions. According to birthday paradox, they are divided into three sorts:*
  - *(1) For hash result, the highest probability is $\frac{q_h^2}{2^{l+1}}$.*
  - *(2) For random numbers α and β, the highest probability is $\frac{(q_s+q_e)^2}{2(n-1)}$.*
  - *(3) For other random numbers such as N_m, N_f, and PID_{MU}^{new}, the highest probability is $\frac{(q_s+q_e)^2}{2^{l+1}}$.*

  So $\mathcal{G}_2$ and $\mathcal{G}_1$ have no difference unless any of the aforementioned cases occurs and $|Pr[Succ_2] - Pr[Succ_1]| \leq \frac{q_h^2+(q_s+q_e)^2}{2^{l+1}} + \frac{(q_s+q_e)^2}{2(n-1)}$.
- *Game $\mathcal{G}_3$: This game includes the cases that $\mathcal{A}$ can forge messages without random oracles.*

– *For Send* $\left(MU^i, FA^j, M_1\right)$*, the simulator checks if* $\{M_1\} \in \mathcal{L}_{\mathcal{P}}$ *and* $(*\|PID_{MU}\|*\|C_3, C_4)$, $(*\|PID_{MU}\|ID_{HA}, *)$, $(ID_{MU}\|*, *) \in \mathcal{L}_{\mathcal{A}}$*, but we should say that* $(PW_{MU}\|b_{MU}, *)$ *cannot be checked. The probability for the first hash query is* $\frac{q_s}{2^l}$*, and each of the other hash queries is* $\frac{q_h}{2^l}$.

– *For Send*(FA^j, HA^t, M_2)*, the simulator checks if* $\{M_1, M_2\} \in \mathcal{L}_{\mathcal{P}}$*, and* $(N_m\|PID_{MU}\|ID_{MU}\|C_3, C_4)$, $(PID_{MU}\|C_1\|C_2\|C_3\|C_4\|C_5\|N_f, C_7)$, $(ID_{MU}\|*, *)$, $(N_m\|PID_{MU}\|ID_{HA}, *)$, $(C_3\|C_5, *) \in \mathcal{L}_{\mathcal{A}}$*. Like the last case,* $(PW_{MU}\|b_{MU}, *)$ *cannot be verified. The probability for each of the first two hash queries is* $\frac{q_s}{2^l}$*, and for each of other queries is* $\frac{q_h}{2^l}$.

– *For Send*(HA^t, FA^j, M_3)*, the simulator checks if* $\{M_1, M_2, M_3\} \in \mathcal{L}_{\mathcal{P}}$*, and* $(C_9\|N_m\|*\|PID_{MU}\|*\|ID_{HA}, C_{12})$, $(K_{fh}\|N_f\|C_3\|C_5\|ID_{FA}\|ID_{HA}, C_{13})$, $(ID_{MU}\|*, *)$, $(*\|ID_{HA}\|*, *)$, $(*\|C_3\|C_5\|*, *) \in \mathcal{L}_{\mathcal{A}}$*, but* $(C_8\|N_m, *)$ *cannot be checked. The probability for each of the first two hash records is* $\frac{q_s}{2^l}$*, and each of the others owns* $\frac{q_h}{2^l}$.

– *For Send*(FA^j, MU^i, M_4)*, the simulator checks if* $\{M_1, M_2, M_3, M_4\} \in \mathcal{L}_{\mathcal{P}}$*, and* $(C_{16}\|N_m\|ID_{MU}\|PID_{MU}\|PID_{MU}^{new}\|ID_{HA}, C_{12})$, $(*\|PID_{MU}\|ID_{HA}, C_{14})$, $(ID_{MU}\|*, C_{15})$, $(PID_{MU}^{new}\|ID_{HA}\|*, C_{16})$, $(C_{15}\|N_m, *)$, $(C_{15}\|C_3\|C_5\|N_m, *) \in \mathcal{L}_{\mathcal{A}}$*. The probability for each of the first two hash queries is* $\frac{q_s}{2^l}$*, and each of the others owns* $\frac{q_h}{2^l}$.

So $\mathcal{G}_3$ *is same as* $\mathcal{G}_2$ *unless any of the forgeries happens. So* $|Pr[Succ_3] - Pr[Succ_2]| \leq \frac{7q_s+15q_h}{2^l}$.

- *Game* $\mathcal{G}_4$*:* $\mathcal{A}$ *can employ the random oracles in this game, and ECDH problem is added. According to* [6,10]*,* $\mathcal{A}$ *should first request the Corrupt*$(MU^i$*, smart card*$)$ *because he could do nothing if he only steals* PW_{MU}*. Two cases are listed:*

  (1) $\mathcal{A}$ *can select one password from the password set and initiate a session. The probability is* $\frac{q_s}{|\mathcal{D}|}$*. This is the active attack.*
  (2) It is the inactive attack. Two sub-cases are listed as follows:

    (a) Execute queries are queried. Finally, $\mathcal{A}$ *must query* $(\alpha P\|\beta P\|\alpha\beta P, SK)$ *to win. The probability that the record can be found in* $\mathcal{L}_{\mathcal{A}}$ *is* $\frac{1}{q_h}$*, so the probability for this sub-case is* $q_h Adv_{\mathcal{A}}^{ECDH}(t+4q_eT_m)$.
    (b) Send queries are asked successively to simulate the Execute query. Like the last sub-case, the probability is $q_h Adv_{\mathcal{A}}^{ECDH}(t + 2q_sT_m)$.

  Let $t' = t + (2q_s + 4q_e)T_m$*, and we can see that* $q_h Adv_{\mathcal{A}}^{ECDH}(t + 4q_eT_m) + q_h Adv_{\mathcal{A}}^{ECDH}(t + 2q_sT_m) \leq 2q_h Adv_{\mathcal{A}}^{ECDH}(t')$.

  Thus, we see that $|Pr[Succ_4] - Pr[Succ_3]| \leq \frac{q_s}{|\mathcal{D}|} + 2q_h Adv_{\mathcal{A}}^{ECDH}(t')$.

- *Game* $\mathcal{G}_5$*: We consider the security character strong forward security here. According to the concept of sfs – fresh,* $Corrupt(\mathcal{I}^k)$ *must be asked after Test, so this game only affects old simulation results. In other words, the answers to the queries of* $\mathcal{A}$ *are selected from old transcripts. Similar to* $\mathcal{G}_4$*, we can see the probability for the existence of* αP*, and* βP *in the same session is* $\frac{1}{(q_s+q_e)^2}$ *and* $|Pr[Succ_5] - Pr[Succ_4]| \leq 2q_h(q_s + q_e)^2 Adv_{\mathcal{A}}^{ECDH}(t')$.

Until now, $\mathcal{A}$ *has no advantage and* $Pr[Succ_5] = \frac{1}{2}$*. Combining all results of the aforementioned games, Theorem 1 is proved.*

7. FORMAL SECURITY VERIFICATION

ProVerif is a popular cryptographic protocol verifier. It proves security features including authentication, strong secrecy, and equivalences between processes. The attacker simulated by the tool owns many strong abilities such as controlling communication channel, and mastering data perhaps appear in the authentication. A formal verification is made with Proverif by us. First, the premises containing channels, shared keys, constants, equations, reductions, and functions are shown in Table V. Then, the events and queries are presented in Table VI. The four events are time points for MU and FA in MAKA phase. The first two queries test the security of session keys, and the last two queries test if right relations exist between the events earlier.

The processes of MU, FA, and HA are illustrated in Tables VII, VIII, and IX, respectively. Here, we simply explain the codes as follows:

- In Table VII, there are two blanks. Lines 2 to 5 of the code in the first blank is for the process of MU registration phase, which is for Steps 1 and 3 in Section 5.1. The second blank is for the Steps 1 and 5 of MAKA phase, in which the user prepares for the message M_1 and deals with the message M_4.
- In Table VIII, there are also two blanks. In the first blank, $in(sch2, (kfh : bitstring))$; denotes the initialization process that the common secret key K_{fh} is received from the HA. The code in the second blank is for the MAKA phase of FA, or Steps 2 and 4 in Section 5.2.

Table V. Premises of code.

```
(*———channels———*)
free ch1: channel.
free ch2: channel.
free sch1: channel [private].
free sch2: channel [private].

(*———shared keys———*)
free skmu: bitstring [private].
free skfa: bitstring [private].

(*——constants——*)
free x:bitstring [private].
free IDMU:bitstring [private].
free PWMU:bitstring [private].
free hkfh:bitstring [private].
const IDFA:bitstring.
const IDHA:bitstring.
const P:bitstring.
table d(bitstring).

(*—functions,reductions and equations—*)
fun h(bitstring):bitstring. (*hash function*)
fun mul(bitstring,bitstring):bitstring. (*scalar multiplication*)
fun add(bitstring,bitstring): bitstring.(*addtion*)
fun xor(bitstring,bitstring):bitstring. (*X-or*)
fun con(bitstring,bitstring):bitstring. (*string concatenation*)
fun senc(bitstring,bitstring):bitstring. (*symmetric encryption*)
reduc forall m:bitstring, n:bitstring; sdec(senc(m,n),n)=m. (*symmetric decryption*)
equation forall m:bitstring,n:bitstring; xor(xor(m,n),n)=m.
equation forall m:bitstring,n:bitstring; mul(m,mul(n,P)) = mul(n,mul(m,P)).
```

Table VI. Events and queries.

(*————event————*)	(*————queries————*)
event MUStart(bitstring).	query attacker(skmu).
event MUAuth(bitstring).	query attacker(skfa).
event FAStart(bitstring).	query id:bitstring; inj-event(MUAuth(id))==> inj-event(MUStart(id)).
event FAAuth(bitstring).	query id:bitstring; inj-event(FAAuth(id))==> inj-event(FAStart(id)).

- In Table IX, there are four parts. The code in the first part is for the registration operations in Section 5.1. The second is for the initialization between HA and FA, and K_{fh} is sent to FA. The third one shows the operations in MAKA phase, or in Section 5.2. The code in last part denotes the whole process of HA. We consider the process *HA* consists of *HAReg*1, *HAReg*2, and *HAAuth*.

Moreover, we use the command line *process* !*MU* | !*HA*| !*FA* to execute the entire codes.

We demonstrate the results here. They mean that the protocol is secure via analysis of Proverif.

RESULT inj – event(*FAAuth*(*id*)) ==> *inj – event*(*FAStart*(*id*)) *is true*.
RESULT inj – event(*MUAuth*(*id*)) ==> *inj – event*(*MUStart*(*id*)) *is true*.
RESULT not attacker(*skfa*[]) *is true*.
RESULT not attacker(*skmu*[]) *is true*.

We see that all results of the queries are "true." The first denotes that the events pairs (*FAStart*, *FAAuth*) are executed in correct orders, or we say that the event *FAAuth* is executed after *FAStart*. Similar as the second result. The third result means that the session key SK_{FA} is secure against the simulated attacks formed by Proverif and so does the session key SK_{MU}.

Table VII. Process of mobile user.

```
let MU=
new bMU:bitstring;
let HPWMU = h(con(PWMU,bMU)) in
out(sch1,(IDMU,HPWMU));
in(sch1,(B1:bitstring,B2:bitstring,PIDMU:bitstring));
```

```
!
(
event MUStart(IDMU);
new alpha:bitstring;
new Nm:bitstring;
let HPWMU = h(con(PWMU,bMU)) in
let C1 = xor(xor(B1,HPWMU),Nm) in
let C2 = xor(h(con(con(Nm,PIDMU),IDHA)),IDMU) in
let C3 = mul(alpha,P) in
let C4 = h(con(con(con(Nm,PIDMU),IDMU),C3)) in
let M1 = (PIDMU,C1,C2,C3,C4) in
out(ch1,M1);
in (ch1,(mC5:bitstring,mC10:bitstring,mC11:bitstring,mC12:bitstring,mC14:bitstring));
let C15 = xor(B2,h(con(IDMU,HPWMU))) in
let C16 = xor(mC10,h(con(C15,Nm))) in
let PIDMUnew = xor(mC11,h(con(con(con(C15,C3),mC5),Nm))) in
if mC12 = h(con(con(con(con(con(C16,Nm),IDMU),PIDMU),PIDMUnew),IDHA)) then
let skmu = h(con(con(C3,mC5),mul(alpha,mC5))) in
if mC14 = h(con(con(skmu,PIDMU),IDHA)) then
let B1new = xor(C16,HPWMU) in
let B1 = B1new in
let PIDMU = PIDMUnew in
0
).
```

Table VIII. Process of foreign agent.

```
let FA =
in(sch2,(kfh:bitstring));
```

```
!
(
in(ch1,(faPIDMU:bitstring,faC1:bitstring,faC2:bitstring,faC3:bitstring,faC4:bitstring));
event FAStart(IDFA);
new beta:bitstring;
new Nf:bitstring;
let C5 = mul(beta,P) in
let C6 = xor(senc(Nf,kfh),h(con(faC3,C5))) in
let C7 = h(con(con(con(con(con(con(faPIDMU,faC1),faC2),faC3),faC4),C5),Nf)) in
let M2 = (faPIDMU,faC1,faC2,faC3,faC4,C5,C6,C7) in
out(ch2,M2);
in(ch2,(faC10:bitstring,faC11:bitstring,faC12:bitstring,faC13:bitstring));
if faC13 = h(con(con(con(con(con(kfh,Nf),faC3),C5),IDFA),IDHA)) then
let skfa = h(con(con(faC3,C5),mul(beta,faC3))) in
let C14 = h(con(con(skfa,faPIDMU),IDHA)) in
let M4 =(C5,faC10,faC11,faC12,C14) in
out (ch1,M4);
0
).
```

Table IX. Process of home agent.

```
let HAReg1 =
in(sch1,(hIDMU:bitstring,hHPWMU:bitstring));
new hPIDMU:bitstring;
insert d(hIDMU);
let hB1 = xor(h(con(con(hPIDMU,IDHA),x)),hHPWMU) in
let hB2 = xor(h(con(hIDMU,x)),h(con(hIDMU,hHPWMU))) in
out (sch1,(hB1,hB2,hPIDMU)).

let HAReg2 =
out(sch2,(hkfh)).

let HAAuth =
in (ch2,(haPIDMU:bitstring,haC1:bitstring,haC2:bitstring,haC3:bitstring,haC4:bitstring,
haC5:bitstring,haC6:bitstring,haC7:bitstring));
let haNm = xor(haC1,h(con(con(haPIDMU,IDHA),x))) in
let haIDMU = xor(haC2,h(con(con(haNm,haPIDMU),IDHA))) in
get d(=haIDMU) in
if haC4 = h(con(con(con(con(haNm,haPIDMU),haPIDMU),haIDMU),haC3)) then
let haNf = sdec(xor(haC6,h(con(haC3,haC5))),hkfh) in
if haC7 = h(con(con(con(con(con(con(haPIDMU,haC1),haC2),haC3),haC4),haC5),haNf)) then
new haPIDMUnew:bitstring;
let C8 = h(con(haIDMU,x)) in
let C9 = h(con(con(haPIDMUnew,IDHA),x)) in
let C10 = xor(C9,h(con(C8,haNm))) in
let C11 = xor(h(con(con(con(C8,haC3),haC5),haNm)),haPIDMUnew) in
let C12 = h(con(con(con(con(con(C9,haNm),haIDMU),haPIDMU),haPIDMUnew),IDHA)) in
let C13 = h(con(con(con(con(con(hkfh,haNf),haC3),haC5),IDFA),IDHA)) in
let M3 = (C10,C11,C12,C13) in
out(ch2,M3).

let HA = HAReg1 | HAReg2 | HAAuth.
```

8. INFORMAL SECURITY ANALYSIS

In this section, we demonstrate the security properties of our scheme and compare our scheme with some recent schemes [28–31] in Table X, according to Section 2.3. If the scheme satisfies the property, we fill the blank with ✓. Otherwise, we use ×.

8.1. Resistant to the insider attack

In the registration phase, MU submits ID_{MU} and HPW_{MU} to HA. A random number b_{MU} is applied to protect PW_{MU} with a hash function, so a malicious inside adversary $\mathcal{A}$, like the administrator of HA, cannot get or guess PW_{MU}, and this attack is avoided.

8.2. Resistant to the off-line guessing attack

Suppose $\mathcal{A}$ retrieves B_1, B_2, B_3, PID_{MU}, and ID_{HA} from MU's smart card and has already got $\{M_1^{old}, M_2^{old}, M_3^{old}, M_4^{old}\}$ from the last session of MU. $\mathcal{A}$ guesses the pair $\left(ID^*, PW^*\right)$ and computes $b^* = B_3 \oplus h(ID^* \| PW^*)$ and $HPW^* = h(PW^* \| b^*)$. Then, two equations can be used by $\mathcal{A}$: $B_1 \oplus HPW^* = C_{10}^{old} \oplus h((B_2 \oplus h(ID^* \| HPW^*)) \| N_m^{old})$ and $PID_{MU} = C_{11}^{old} \oplus h((B_2 \oplus h(ID^* \| HPW^*)) \| C_3^{old} \| C_5^{old} \| N_m^{old})$. But N_m^{old} is the random number produced by MU in last session, and the only equation can be used by $\mathcal{A}$ is $B_1^{old} \oplus HPW^* \oplus N_m^{old} = C_1^{old}$. Unfortunately $\mathcal{A}$ cannot gain B_1^{old}, which has already disappeared, so this attack can be avoided.

8.3. Resistant to the user forgery attack

If $\mathcal{A}$ wants to forge M_1, he has to calculate $C_1 = h(PID_{MU} \| ID_{HA} \| x) \oplus N_m$. However, x is a secret key stored in HA, and $\mathcal{A}$ has no ability to obtain it. This means the user forgery attack cannot be finished.

8.4. Resistant to the foreign agent and home agent forgery attacks

In order to forge M_2, it is hard for $\mathcal{A}$ to produce a legal C_6 without knowing the secret key K_{fh}. Similarly, to forge C_{10}, C_{11}, C_{12}, and C_{13}, $\mathcal{A}$ should master x and K_{fh}, so FA and HA forgery attacks cannot be carried out.

DOI: 10.1002/sec

Table X. Security properties comparison.

Characters	[30]	[31]	[29]	[28]	Ours
Resistant to the insider attack	✓	✓	✓	✓	✓
Resistant to the off-line guessing attack	✓	×	×	×	✓
Resistant to the user forgery attack	✓	✓	✓	✓	✓
Resistant to the foreign agent forgery attack	✓	✓	✓	✓	✓
Resistant to the home agent forgery attack	✓	✓	✓	✓	✓
Resistant to the de-synchronization attack	×	×	✓	✓	✓
Resistant to the replay attack	✓	✓	✓	✓	✓
Resistant to the known-key attack	✓	✓	✓	✓	✓
Session key unknown to home agent	×	×	×	×	✓
User anonymity	✓	✓	✓	×	✓
Mutual authentication	✓	✓	✓	✓	✓
Strong forward security	✓	×	×	×	✓

8.5. Resistant to the de-synchronization attack

In Section 5.3, the old password must be verified by HA, and then, the new password change request is permitted. The way in our scheme keeps MU from the hazard of careless password input, and of course, from the de-synchronization attack.

8.6. Resistant to the replay attack

In every session, MU and FA generate new random numbers α, β, N_m, and N_f. If $\mathcal{A}$ replays the old message M_1 including $C_3^{old} = \alpha^{old}P$ to FA, FA will produce a new $\beta^{new} \in Z_n^*$. At last, $\mathcal{A}$ cannot calculate the correct SK_{MU} to go on the latter conversation because he does not know α^{old}, or he faces the elliptic curve discrete logarithm problem, so this attack can be kept away.

8.7. Resistant to the known-key attack

If $\mathcal{A}$ obtains some session keys, the other session keys will not be affected. There is no relation between any pair of the session keys because each session the user and FA generate new α and β.

8.8. Session key unknown to home agent

The session keys in our scheme are based on ECDH problem. There is not any contribution from HA for them, so any calculation in HA is helpless with the final session key, and the malicious administrator cannot know the session key.

8.9. User anonymity

In our scheme, MU sends a new pseudo-identity PID_{MU}, which is only a random string in each session, and ID_{MU} is protected by a hash result $h(N_m||PID_{MU}||ID_{HA})$ with exclusive-or computation. In Section 8.2, we have explained the random number N_m cannot be computed by $\mathcal{A}$, so our scheme keeps users anonymous, and the attacker cannot track the identity of U_i with no doubt.

8.10. Mutual authentication

Among the messages, HA checks C_4 and C_7 to authenticate MU and FA, respectively. FA verifies C_{13} to authenticate HA directly and MU indirectly, and MU verifies C_{12} and C_{14} to authenticate HA and FA, respectively, so all three entities reach mutual authentication.

8.11. Strong forward security

Even if $\mathcal{A}$ obtains all long-term data of the three entities referred in the session, including the critical secret strings such as x, K_{fh} and PW_{MU}, he cannot calculate any past session key SK_{MU} or SK_{FA}. The reason is that SK_{MU} and SK_{FA} are based on ECDH problem.

9. PERFORMANCE EVALUATION

Here, we compare our scheme with the recent schemes [28–31] mentioned in Table X. At first, the platform for time cost test and some basic parameters including lengths and time are illustrated in Table XI. Moreover, we consider the length of the random numbers is 160 bits.

We make the comparison of MAKA phase for five schemes in Table XII and explain the results as follows:

- Our scheme only takes less time than [29] on the aspects of MU and FA, and less than [29,31] on the aspect of HA. The reason is that ECC is employed on MU and FA in our scheme and is also in Zhang *et al.* scheme [29].
- For the communication cost, our scheme is only better than [29]. The reason is like the aforementioned item. The cost of points in G takes a large part of the whole transmitted messages.

Table XI. Parameters of referred cryptographic operations.

CPU: Intel(R) Core TM i7-4710HQ 2.50GHz Memory:8G OS:Win8 64-bit Software: Visual C++ 2010,MIRACL C/C++ Library Security level: 160-bit point in F_p, AES(every block 128bits),SHA2-256		
Symbols	Meaning	Time cost (ms)
T_m	Time cost of one scalar multiplication	0.427576
T_s	Time cost of one AES operation	0.0214835
T_h	Time cost of one hash	0.005174

Table XII. Performance comparison.

	[30]	[31]	[29]	[28]	Ours
Time cost of MU (ms)	$6T_h$ = 0.031044	$T_s + 5T_h$ = 0.043535	$2T_m + 2T_s + 2T_h$ = 0.908467	$6T_h$ = 0.031044	$2T_m + 10T_h$ = 0.906892
Time cost of FA (ms)	$5T_h$ = 0.02587	$2T_s + 2T_h$ = 0.068837	$2T_m + 2T_s + 2T_h$ = 0.908467	$2T_s + T_h$ = 0.048141	$2T_m + T_s + 5T_h$ = 0.9025055
Time cost of HA (ms)	$10T_h$ = 0.05174	$3T_s + 7T_h$ = 0.1006685	$5T_s + T_h$ = 0.1125915	$2T_s + 5T_h$ = 0.068837	$T_s + 11Th$ = 0.078975
Communication cost (bits)	4032	4384	6240	2944	5696
Formal proof	No	No	Yes	Yes	Yes
Formal verification	Yes	No	No	No	Yes
Security and practicality	No	No	No	No	Yes

MU, mobile user; FA, foreign agent.

- Only our scheme has both formal proof and formal verification to analyze the security. Others are destitute of one way at least.
- For the last and most important index, only our scheme reaches all the security requirements, while each of others has security weaknesses or is impractical for application. Here, we should say that in Gope and Hwang's scheme [30], if the message from HA to FA is abandoned only because of time delay, the shared secret key K_{fh} needs to be adjusted via a secure channel. Such way is not applicable.

10. CONCLUSION

In this paper, we show that Gope and Hwang's authentication scheme for GLOMONETs is insecure. To diminish the weaknesses in all mentioned recent schemes, a novel mutual authentication scheme is proposed. By showing formal proof, formal verification, and informal security analysis, our scheme is secure and more suitable to put into practice.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their valuable comments. This research is supported by Fujian Education and Scientific Research Program for Young and Middle-aged Teachers under Grant No. JA14369, University Distinguished Young Research Talent Training Program of Fujian Province(Year 2016), and the National Natural Science Foundation of China under Grant No. 61300220. The authors also extend their sincere appreciations to the Deanship of Scientific Research at King Saud University for its funding to this Prolific Research Group (PRG-1436-16).

REFERENCES

1. Wu F, Xu L. Security analysis and improvement of a privacy authentication scheme for telecare medical information systems. *Journal of Medical Systems* 2013; **37**(4), Article ID: 9958.
2. Li X, Ma J, Wang W, Xiong Y, Zhang J. A novel smart card and dynamic id based remote user authentication scheme for multi-server environments. *Mathematical and Computer Modelling* 2013; **58**(1): 85–95.
3. Li X, Niu J, Khan MK, Liao J. An enhanced smart card based remote user password authentication scheme. *Journal of Network and Computer Applications* 2013; **36**(5): 1365–1371.
4. Jiang Q, Ma J, Li G, Yang L. An efficient ticket based authentication protocol with unlinkability for wireless access networks. *Wireless Personal Communications* 2014; **77**(2): 1489–1506.
5. Jiang Q, Ma J, Lu X, Tian Y. An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks. *Peer-to-Peer Networking and Applications* 2015; **8**(6): 1070–1081.

6. Xu L, Wu F. An improved and provable remote user authentication scheme based on elliptic curve cryptosystem with user anonymity. *Security and Communication Networks* 2015; **8**(2): 245–260.
7. Wu F, Xu L, Kumari S, Li X. A novel and provably secure biometrics-based three-factor remote authentication scheme for mobile client cserver networks. *Computers & Electrical Engineering* 2015; **45**: 274–285.
8. Xu L, Wu F. Cryptanalysis and improvement of a user authentication scheme preserving uniqueness and anonymity for connected health care. *Journal of Medical Systems* 2015; **39**(2): 1–9.
9. Wu F, Xu L, Kumari S, Li X. An improved and anonymous two-factor authentication protocol for health-care applications with wireless medical sensor networks. *Multimedia Systems* 2015. DOI:10.1007/s00530-015-0476-3.
10. Wu F, Xu L, Kumari S, Li X. A new and secure authentication scheme for wireless sensor networks with formal proof. *Peer-to-Peer Networking and Applications* 2015. DOI:10.1007/s12083-015-0404-5.
11. Ren Y, Shen J, Wang J, Han J, Lee S. Mutual verifiable provable data auditing in public cloud storage. *Journal of Internet Technology* 2015; **16**(2): 317–323.
12. Wu F, Xu L, Kumari S, Li X. A privacy-preserving and provable user authentication scheme for wireless sensor networks based on internet of things security. *Journal of Ambient Intelligence and Humanized Computing* 2016. DOI:10.1007/s12652-016-0345-8.
13. Jiang Q, Khan MK, Lu X, Ma J, He D. A privacy preserving three-factor authentication protocol for e-health clouds. *The Journal of Supercomputing* 2016, DOI:10.1007/s11227-015-1610-x.
14. Jiang Q, Wei F, Fu S, Ma J, Li G, Alelaiwi A. Robust extended chaotic maps-based three-factor authentication scheme preserving biometric template privacy. *Nonlinear Dynamics* 2016; **83**(4): 2085–2101.
15. Guo P, Wang J, Geng Xue Hua, Kim Chang Seob, Kim Jeong-Uk. A variable threshold-value authentication architecture for wireless mesh networks. *Journal of Internet Technology* 2014; **15**(6): 929–935.
16. Wu F, Xu L. An improved and provable self-certified digital signature scheme with message recovery. *International Journal of Communication Systems* 2015; **28** (2): 344–357.
17. Zhu J, Ma J. A new authentication scheme with anonymity for wireless environments. *IEEE Transactions on Consumer Electronics* 2004; **50** (1): 231–235.
18. Lee CC, Hwang MS, Liao IE. Security enhancement on a new authentication scheme with anonymity for wireless environments. *IEEE Transactions on Industrial Electronics* 2006; **53**(5): 1683–1687.
19. Wu CC, Lee WB, Tsaur WJ. A secure authentication scheme with anonymity for wireless communications. *IEEE Communications Letters* 2008; **12**(10): 722–723.
20. Chang CC, Lee CY, Chiu YC. Enhanced authentication scheme with anonymity for roaming service in global mobility networks. *Computer Communications* 2009; **32**(4): 611–618.
21. Youn TY, Park YH, Lim J. Weaknesses in an anonymous authentication scheme for roaming service in global mobility networks. *IEEE Communications Letters* 2009; **13**(7): 471–473.
22. He D, Ma M, Zhang Y, Chen C, Bu J. A strong user authentication scheme with smart cards for wireless communications. *Computer Communications* 2011; **34** (3): 367–374.
23. Mun H, Han K, Lee YS, Yeun CY, Choi HH. Enhanced secure anonymous authentication scheme for roaming service in global mobility networks. *Mathematical and Computer Modelling* 2012; **55**(1): 214–222.
24. Kim JS, Kwak J. Secure and efficient anonymous authentication scheme in global mobility networks. *Journal of Applied Mathematics* 2013; **2013**, Article ID: 302582.
25. Jiang Q, Ma J, Li G, Yang L. An enhanced authentication scheme with privacy preservation for roaming service in global mobility networks. *Wireless Personal Communications* 2013; **68**(4): 1477–1491.
26. Shin S, Yeh H, Kim K. An efficient secure authentication scheme with user anonymity for roaming user in ubiquitous networks. *Peer-to-peer Networking and Applications* 2015; **8**(4): 674–683.
27. Wen F, Susilo W, Yang G. A secure and effective anonymous user authentication scheme for roaming service in global mobility networks. *Wireless Personal Communications* 2013; **73**(3): 993–1004.
28. Farash MS, Chaudhry SA, Heydari M, Sadough S, Mohammad S, Kumari S, Khan MK. A lightweight anonymous authentication scheme for consumer roaming in ubiquitous networks with provable security. *International Journal of Communication Systems* 2015. DOI:10.1002/dac.3019.
29. Zhang G, Fan D, Zhang Y, Li X, Liu X. A privacy preserving authentication scheme for roaming services in global mobility networks. *Security and Communicaton Networks* 2015; **8**: 2850–2859.
30. Gope P, Hwang T. Lightweight and energy-efficient mutual authentication and key agreement scheme with user anonymity for secure communication in global mobility networks. *IEEE Systems Journal* 2015. DOI:10.1109/JSYST.2015.2416396.

31. Gope P, Hwang T. An efficient mutual authentication and key agreement scheme preserving strong anonymity of the mobile user in global mobility networks. *Journal of Network and Computer Applications* 2016; **62**: 1–8.
32. Wang D, He D, Wang P, Chu CH. Anonymous two-factor authentication in distributed systems: certain goals are beyond attainment. *IEEE Transactions on Dependable and Secure Computing* 2015; **12** (4): 428–442.
33. Kocher P, Jaffe J, Jun B. Differential power analysis. In *Advances in Cryptology-CRYPTO'99*. Springer Berlin Heidelberg: Santa Barbara, California, USA, 1999; 388–397.
34. Messerges TS, Dabbish EA, Sloan RH. Examining smart-card security under the threat of power analysis attacks. *IEEE Transactions on Computers* 2002; **51**(5): 541–552.
35. Mangard S, Oswald E, Standaert FX. One for all call for one: unifying standard differential power analysis attacks. *IET Information Security* 2011; **5** (2): 100–110.
36. Bresson E, Chevassut O, Pointcheval D. Security proofs for an efficient password-based key exchange. *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ACM New York, Washington, DC, USA, 2003; 241–250.