Received August 13, 2016, accepted September 9, 2016, date of publication September 28, 2016,
date of current version October 31, 2016.

Digital Object Identifier 10.1109/ACCESS.2016.2613442

TTP Based High-Efficient Multi-Key Exchange Protocol

KUN-LIN TSAI[1], (Member, IEEE), YI-LI HUANG[2], (Member, IEEE), FANG-YIE LEU[2], (Member, IEEE), AND ILSUN YOU[3], (Senior Member, IEEE)
[1]Department of Electrical Engineering, Tunghai University, Taichung 40704, Taiwan
[2]Department of Computer Science, Tunghai University, Taichung 40704, Taiwan
[3]Department of Information Security Engineering, Soonchunhyang University, Asan 31538, South Korea

Corresponding author: I. You (ilsunu@gmail.com)

This work was supported in part by the Soonchunhyang University Research Fund and in part by the Ministry of Science and Technology, Taiwan, under Grant NSC 101-2221-E-009-003-MY3.

ABSTRACT With a trusted-third-party (TTP)-based key exchange protocol, when a user would like to transmit a message to another user, the transmitted data are encrypted by a session key exchanged between the two ends of the corresponding connection with the help of the TTP. Up to present, due to the assistance of a TTP, this type of protocols has performed well in protecting messages delivered between two authorized users. Even this, inflexibility, unreliability, and inefficiency problems still exist in these previously proposed protocols. Therefore, in this paper, a multi-key exchange protocol, named the TTP-based high-efficient multi-key exchange protocol (THMEP), is proposed to provide users with a secure and efficient protocol, which employs the elliptic curve cryptography, a 2-D operation, and a current time encryption key, to exchange their session keys. The proposed protocol not only effectively hides important encryption parameters, but also achieves fully mutual authentication between a user and his/her trusted server. It can resist known-key, impersonation, replay, eavesdropping, and forgery attacks. Besides, the THMEP generates 40 session keys in a key exchange process, meaning the proposed protocol can support 40 sessions simultaneously. It also shortens the processing time, which is 3.78 times faster than that of a specific previous study. Its security level and performance are higher than those of the compared state-of-the-art protocols. In other words, the THMEP is very suitable for IoT applications.

INDEX TERMS Multiple key exchange, trusted third party, elliptic curve cryptosystem, two-dimensional operation, current time encryption key.

I. INTRODUCTION

In the past decades, with the fast development of wireless communication techniques and mobile services, numerous commercial systems [1], [2] and e-commerce applications [3], [4] have been proposed for users to communicate and share their information with other people. However, owing to the insecure nature of wireless channels, many security issues, such as data leakage and personal privacy, need to be carefully addressed when a wireless communication system is being developed and used.

In wireless communication networks, cryptography is often used to protect users' secret data and guarantee integrity of the data. Asymmetric cryptography methods, like RSA encryption algorithm [5], are often adopted to encrypt delivered messages [6], [7]. However, the computational costs of these algorithms are relatively high due to employing long keys and complex encryption/decryption processes. When they are installed in a mobile device, like a smart phone, encrypting and decrypting data often consumes considerable energy and a long time. The Diffie-Hellman key exchange protocol [8] is another choice which helps users to establish one or several common secret keys, with which to encrypt and decrypt transmitted messages with a symmetric method.

In fact, many key exchange protocols have been proposed [9]–[12]. Unfortunately, each of them has its own weakness in security and performance. For example, the multiplication of an enormous number of large prime number during the encryption/decryption process consumes a long computational time. Besides, in the key-exchange authentication process, some parameters are static, i.e., they remain unchanged throughout the process. This, in fact, significantly decreases the key exchange security, and implies that the protocol could not defend against eavesdropping attack and replay attack [13]. In particular, Li*et al.* [11] proposed a useful three-party password-authenticated multiple key exchange protocol for wireless mobile networks, and claimed that it had high efficiency, reliability, flexibility, and scalability. Li's protocol establishes multiple session keys in its key exchanging process and greatly reduces the server's and users' computational costs. Nevertheless, it is still insecure and inefficient due to employing static keys and a complex process.

Therefore, in this paper, we design a multi-key exchange scheme, named the Trusted-third-party-based High-efficient Multi-Key Exchange Protocol (THMEP for short), which provides users with a high-efficient, reliable, and scalable key exchange method for data communication. Unlike certificate authority (CA) issuing digital certificates to users, the Trusted-third-party (TTP for short) is used to authenticate the users and their messages so that two users can safely exchange important parameters.

In the THMEP, a dynamic parameter is derived from a TTP system's clock to encrypt an important exchanged key so as to prevent eavesdropping and replay attacks. Besides, the THMEP mainly uses a two-dimensional operation (including the logical XOR and a binary adder), and reduces the times of invoking the Elliptic Curve Cryptography (ECC) multiplications, when encrypting important parameters. The purpose is to enhance its performance since the computational cost of an XOR operation or a binary addition is much lower than that of an ECC multiplication. We will show this later. Our simulation result demonstrates that the time the THMEP consumes is much shorter than the time required by Li's key exchange protocol [11]. Our security analysis shows that the THMEP has a higher security level than those of three state-of-the-art approaches, including replay attack prevention, eavesdropping attack prevention, and forgery attack prevention. The contributions of the THMEP are as follows.

1. **Efficiency:** Based on the discrete logarithm, the ECC can achieve a higher security level than that of the RSA [14] when their key lengths are the same. Further, a binary addition with a logical XOR operation consumes shorter time than that spent by an ECC operation. Namely, substituting some ECC operations by the binary addition and XOR operation can further reduce the users' and TTP's computation and communication costs. On the other hand, many communication process of the THMEP can be executed in a parallel manner, resulting in a shorter processing time. Furthermore, the computational complexity of the THMEP is much lower than those of existing protocols. So the energy consumption for its key computation is also reduced, thus very suitable for mobile devices powered by batteries.
2. **Reliability:** The THMEP can protect against many conventional attacks, like replay, eavesdropping, known-key, impersonation, and forgery attacks. The THMEP also provides full mutual authentication between users and the TTP.
3. **High throughput:** A total of 40 session keys are generated in a key exchange process. As a result, the users can create up to 40 individual channels for their following communications.

The remainder of this paper is structured as follows. In Section 2, we introduce the preliminaries and the notations used. In Section 3, Li's 3MPAKE protocol [11] is briefly reviewed. The THMEP protocol is detailed in Section 4. The security and performance analyses of the THMEP are presented and evaluated in Sections 5 and 6, respectively. In Section 7, we conclude the study and outlines our future studies.

II. PRELIMINARIES

A. RELATED WORKS

Bellovin and Merritt [15] in 1992 proposed an encrypted key exchange (EKE) protocol which integrated a secret key and a public key created to prevent delivered data from dictionary attacks. Ballare *et al.* [16] presented a method for the password-based authenticated key exchange (AKE) protocol which shows the correctness of Bellovin and Merritt's idea. Abdalla and Pointcheval [17] also demonstrated a two-password-based encrypting key exchange protocol which is more efficient than the one introduced in [15]. However, when two users are communicating with each other, the shared password may be enumerated by hackers by using dictionary attacks. Sui *et al.* [18] and Lo *et al.* [19] modified the AKE protocols to improve the effectiveness of Abdalla and Pointcheval's protocol. Sui *et al.* [18] defined a two-party authenticated key exchange (2PAKE) protocol which employs the ECC method, and Lo *et al.* [19] proposed a 2PAKE protocol for wireless networks following the specifications of the 3GPP2. However, their passwords are only chosen from a small space, and their protocols request each pair of users sharing a password, causing the fact that a huge amount of passwords are necessary when many users are involved in such a system.

On the other hand, [20] proposed the three-party password-authenticated key exchange (3PAKE) protocol to enhance the security of the 2PAKE. In an insecure network, the 3PAKE utilizes a three-party server to help the communication parties to authenticate each other and exchange session keys. Following the approach proposed in [20], Lu *et al.* [21] introduced a simple three-party password based authenticated key exchange (S-3PAKE) protocol to eliminate server's public key. However, in [22], Chung *et al.* showed that the S-3PAKE is still exposed to impersonation-of-initiator attack, and they used a counter to resist such attack. Nevertheless, [23] indicated that the protocols claimed in [21] and [22] are still vulnerable to man-in-the-middle attack and unknown key-share attack. In order to reduce the communication steps of 3PAKE, [24] designed an efficient 3PAKE protocol requiring neither server public key, nor symmetric cryptosystems. Meanwhile, [25] presented several proposed protocols which are still vulnerable to attacks, such as undetectable online-dictionary attacks [26], key-share attacks [27], and both online and offline password guessing attacks [28].

Up to present, several studies have been proposed to lower the computation cost and enhance the efficiency of the 3PAKE. Reference [29] pointed out that they reduce communication latency, remove the table for storing keys, and lower the requirement of computational resources. They are essential in improving the efficiency and security of the 3PAKE protocol. However, Yang and Chang [30] figured out that the limitations on communication bandwidth and energy consumption of the 3PAKE have shown unsuitable for wireless mobile networks, hence proposing an efficient three-party authenticated key exchange protocol based on the ECC, and claimed that their protocol has a lower computation cost and lighter communication loads than the 3PAKE has in [31]. In 2012, Li *et al.* [11] develop a three-party password-authenticated multiple key exchange (3MPAKE) protocol for wireless mobile networks, and claimed that it had higher efficiency, reliability, flexibility, and scalability than those of the 3PAKE. The 3MPAKE protocol establishes multiple session keys in its key exchanging process, thus greatly reducing the server's and users' computational costs. Comparing the 3MPAKE with the 3PAKE, the server's computation load of the former is lower when the number of users is large, meaning it is one with better scalability and the server can serve many more users at the same time without dramatically degrading its service performance. Besides, some communication steps in the 3MPAKE are executed in parallel. This further effectively shortens its required execution time.

B. DIFFIE-HELLMAN KEY EXCHANGE

The Diffie-Hellman key exchange algorithm [6], which was first published in 1976, enables two users to securely exchange a key to be used in the subsequent message encryption.

In this algorithm, there are two publicly known numbers: a prime number q and an integer α, in which the latter is a primitive root of q. Assume that the users A and B wish to exchange a key. User A selects a random integer $X_A < q$ and computes $Y_A \equiv \alpha^{X_A} \bmod q$. Similarly, user B independently selects a random integer $X_B < q$ and computes $Y_B \equiv \alpha^{X_B} \bmod q$. Each side keeps the X value private (i.e., A (B) keeps $X_A(X_B)$), and sends the Y value (i.e., Y_A and Y_B) to the other side. User A computes the key as $K \equiv (Y_B)^{X_A} \bmod q$, and user B computes the key as $K \equiv (Y_A)^{X_B} \bmod q$. These two equations produce identical results:

$$\begin{aligned} K &\equiv (Y_B)^{X_A} \bmod q \\ &\equiv \alpha^{X_B X_A} \bmod q \\ &\equiv (Y_A)^{X_B} \bmod q \end{aligned}$$

The security of the Diffie-Hellman key exchange process heavily relies on the difficulty of calculating discrete logarithms [31]. For large primes, it is considered infeasible.

Key exchange using elliptic curves can be done in the following manner. Let G be a cyclic additive group derived from a public point P, whose order is a prime n. The problems in the additive group $(G, +)$ are as follows.

(1) Elliptic curve discrete logarithm problem: Given two group elements P and Q, it is difficult to find an integer $x \in Z_n^*$, such that $Q \equiv x \cdot P$ whenever such an integer exists.
(2) Elliptic curve decision Diffie-Hellman problem: For $a, b, c \in Z_n^*$, given p, aP, bP, and cP (P is the generator), it is hard to decide whether $c \equiv ab \bmod q$.
(3) Elliptic curve computational Diffie-Hellman problem: For $a, b, c \in Z_n^*$, gives p, aP, and bP, it is difficult to compute abP.

C. NOTATIONS

The notations used in this study are illustrated and defined in Table 1.

III. REVIEW OF 3MPAKE PROTOCOL

In this section, we briefly review Li's protocol [11], i.e., the 3MPAKE, which consists of two phases, including the initialization phase and the authentication and key exchange phase.

A. THE INITIALIZATION PHASE

In this phase, a trusted server S (or simply S or server S) generates system parameters. First, server S selects a finite field F_q over a large prime number q, where $q > 2^w$, and w is the chosen key size. An elliptic curve equation E: $y^2 \equiv x^3 + ax + b \pmod q$ over F_q is determined, where $a, b \in F_q$, and it satisfies the condition of $4a^3 + 27b^2 \neq 0 \pmod q$. Meanwhile, server S selects a public point P with the order n over E, and utilizes P to generate a cyclic additive group G of order n over E. S further chooses three hash functions $H_1 : U^2 \times D \rightarrow Z_n^*$, $H_2 : U^3 \times G^4 \rightarrow Z_n^*$, and $H_3 : U^3 \times G^4 \rightarrow Z_n^*$ where $U = \{0, 1\}^*$ and D is a password space with a finite number of passwords. S continues selecting a secure pseudo-random function F, and then publishes the system parameters $\{G, P, H_1, H_2, H_3, F\}$.

Assume that users A and B would like to join the system. They choose their own passwords pw_A and pw_B from the password space D, compute $v_A = H_1(ID_A, ID_S, pw_A)$ and $v_B = H_1(ID_B, ID_S, pw_B)$ individually, and share the verifiers $V_A \equiv v_A P$ and $V_B \equiv v_B P$ with the server.

B. THE AUTHENTICATION AND KEY EXCHANGE PHASE

Based on the help of the server S, users A and B in this phase could authenticate each other and generate session keys for the following communication.

Round 1:

Server S

(1.1) produces two random numbers s_A and s_B where $s_A, s_B \in Z_n^*$;
(1.2) computes $S_A \equiv s_A P$; $S_B \equiv s_B P$; $S_A^* \equiv S_A + V_A$; $S_B^* \equiv S_B + V_B$;

TABLE 1. Notations used in this study and 3MPAKE.

Notations		Descriptions
3MPAKE	THMEP	
E	E	An elliptic curve equation
G	G	The elliptic curve group over E
P	P	A public point with order n (generator of G)
H_1, H_2, H_3	H_1, H_2	One-way hash functions
U	U	Bit string of 0 and 1, i.e., {0,1}*
F	-	A secure pseudo-random function
F_q	F_q	A finite field over a large prime number q
D	D	Password space with a finite number of passwords
pw_A , pw_B	pw_A , pw_B	Passwords of user A and user B
ID_S, ID_A , ID_B	ID_S, ID_A , ID_B	IDs of the trusted server S, user A, and user B
q, n	q, n	Prime numbers
s, s_A, s_B	s_A, s_B	Random numbers generated by the trusted server S
x_{A1}, x_{A2}, k_A	x_A	Random numbers generated by user A
x_{B1}, x_{B2}, k_B	x_B	Random numbers generated by user B
S_A, S_B	S_A, S_B	Encrypted random numbers generated by the trusted server S
r_A, y_A	X_A	Authentication parameter(s) of user A
r_B, y_B	X_B	Authentication parameter(s) of user B
τ_A, τ_B	η_A, η_B	Authentication parameters of the trusted server S
SK_i	SK_i	Session keys between user A and user B
v_A, v_B	k_{PWA}, k_{PWB}	Password keys of user A and user B
$K_0 \sim K_3$	$K_0 \sim K_3$	Encrypted cross-paired parameters of user A and user B
V_A, V_B	-	Verifiers of user A and user B
e_A, e_B	-	Evidence parameters of user A and user B
S_A*, S_B*	-	Random numbers secured by the verifiers of user A and user B
X_{A1}, X_{A2}	-	Encrypted random numbers generated by user A
X_{B1}, X_{B2}	-	Encrypted random numbers generated by user B
x	-	The x-coordinate of an elliptic point
K_{AS}, K_{BS}	-	The trusted server's keys used to authenticate user A and user B
K_{SA}, K_{SB}	-	User's keys used to authenticate the trusted server
-	$t_{nonce,S}$, $t_{nonce,A}$, $t_{nonce,B}$	System times of the trusted server S, user A and user B
-	k_{CT}	Current-time encryption key
-	s'_A, s'_B	Encrypted random numbers of the trusted server
-	Δt	Time threshold
-	$s_{A,C}$, $s_{B,C}$	Computed random numbers by user A and user B
-	$S_{A,C}$, $S_{B,C}$	Computed user keys used to authenticate user A and user B
-	x'_A, x'_B	Encrypted random numbers of user A and user B
-	$x_{A,C}$, $x_{B,C}$	Computed random numbers of the trusted server
-	$X_{A,C}$, $X_{B,C}$	Computed server keys used to authenticate user A and user B

(1.3) delivers message ID_S, S_A^* to A and message ID_S, S_B^* to B.

Round 2:

On receiving the message, user A

(2.1a) produces three random numbers x_{A1}, x_{A2}, and k_A where $x_{A1}, x_{A2}, k_A \in Z_n^*$;

(2.2a) calculates $S_A \equiv S_A^* - V_A$; $X_{A_1} \equiv x_{A1}P$; $X_{A2} \equiv x_{A2}P$; $K_{AS} \equiv (x_{A1} + x_{A2})S_A$ and $e_A = H_2(ID_A, ID_S, ID_B, X_{A1}, X_{A2}, S_A^*, K_{AS})$;

(2.3a) computes $r_A \equiv (k_AP)x \bmod n$ and $y_A \equiv k_A^{-1}(e_A + v_Ar_A) \bmod n$, where k_A^{-1} is calculated by representing k_A as a matrix and then computes k_A's inverse matrix as k_A^{-1};

(2.4a) transmits $ID_A, X_{A1}, X_{A2}, r_A, y_A$ to server S.

Similarly and simultaneously, user B

(2.1b) produces three random numbers x_{B1}, x_{B2}, and k_B where $x_{B1}, x_{B2}, k_B \in Z_n^*$;

(2.2b) calculates $S_B \equiv S_B^* - V_B$; $X_{B1} \equiv x_{B1}P$; $X_{B2} \equiv x_{B2}P$; $K_{BS} \equiv (x_{B1} + x_{B2})S_B$ and $e_B = H_2(ID_B, ID_S, ID_A, X_{B1}, X_{B2}, S_B^*, K_{BS})$;

(2.3b) computes $r_B \equiv (k_BP)x \bmod n$ and $y_B \equiv k_B^{-1}(e_B + v_Br_B) \bmod n$;

(2.4b) transmits $\{ID_B, X_{B1}, X_{B2}, r_B, y_B\}$ to server S.

Round 3:

When individually receiving the two transmitted messages from A and B, server S

(3.1) calculates $K_{AS} \equiv s_A(X_{A1} + X_{A2})$; $e_A = H_2(ID_A, ID_S, ID_B, X_{A1}, X_{A2}, S_A^*, K_{AS})$ and $R_A \equiv y_A^{-1}(e_AP + r_AV_A)$;

(3.2) verifies whether or not $xR_A \bmod n \equiv r_A$; if not, server S terminates the session. Otherwise, it

(3.3) computes $K_{BS} \equiv s_B(X_{B1} + X_{B2})$; $e_B = H_2\left(ID_B, ID_S, ID_A, X_{B1}, X_{B2}, S_B^*, K_{BS}\right)$ and $R_B \equiv y_B^{-1}(e_BP + r_BV_B)$;

(3.4) checks to see whether or not $xR_B \bmod n \equiv r_B$; if not, server S terminates the session. Otherwise, it

(3.5) produces a random number s, $s \in Z_n^*$;

(3.6) calculates $Z_{A1} \equiv sX_{B1}$; $Z_{A2} \equiv sX_{B2}$; $K_{SA} \equiv s_A(X_{A1} + X_{A2} + V_A)$; $Z_{B1} \equiv sX_{A1}$; $Z_{B2} \equiv sX_{A2}$; $K_{SB} = s_B(X_{B1} + X_{B2} + V_B)$; $\tau_A = H_3(ID_A, ID_S, ID_B, X_{A1}, X_{A2}, S_A, K_{SA})$ and $\tau_B = H_3(ID_B, ID_S, ID_A, X_{B1}, X_{B2}, S_B, K_{SB})$;

(3.7) delivers message $\{ID_S, ID_B, Z_{A1}, Z_{A2}, \tau_A\}$ to A and $\{ID_S, ID_A, Z_{B1}, Z_{B2}, \tau_B\}$ to B.

Round 4:

Upon receiving the message $\{ID_S, ID_B, Z_{A1}, Z_{A2}, \tau_A\}$, user A

(4.1a) calculates $K_{SA} = (x_{A1} + x_{A2} + v_A)S_A$;

(4.2a) checks to see whether or not $\tau_A = H_3(ID_A, ID_S, ID_B, X_{A1}, X_{A2}, S_A, K_{SA})$; if not, user A terminates this session. Otherwise, it

(4.3a) calculates $K_0 \equiv x_{A1}Z_{A1}$; $K_1 \equiv x_{A2}Z_{A1}$; $K_2 \equiv x_{A1}Z_{A2}$ and $K_3 \equiv x_{A2}Z_{A2}$.

(4.4a) produces session key $SK_i = F_{K_i}(ID_A, ID_S, ID_B)$ in which $i \in \{0, 1, 2, 3\}$.

Similarly, when receiving the message $\{ID_S, ID_A, Z_{B1}, Z_{B2}, \tau_B\}$, user B

(4.1b) calculates $K_{SB} = (x_{B1} + x_{B2} + v_B)S_B$;

(4.2b) checks to see whether or not $\tau_B = H_3(ID_B, ID_S, ID_A, X_{B1}, X_{B2}, S_B, K_{SB})$; if not, it terminates the session. Otherwise, it

(4.3b) calculates $K_0 \equiv x_{B1}Z_{B1}$; $K_1 \equiv x_{B1}Z_{B2}$; $K_2 \equiv x_{B2}Z_{B1}$ *and* $K_3 \equiv x_{B2}Z_{B2}$.

(4.4b) produces session key $SK_i = F_{K_i}(ID_A, ID_S, ID_B)$ in which $i \in \{0, 1, 2, 3\}$.

IV. THE THMEP

The THMEP also consists of two phases, namely, the initialization phase and the authentication and key exchange phase. The binary operation used in this protocol is first introduced.

A. BINARY ADDITION/SUBTRACTION

Analogous to the addition on natural numbers, a binary adder adds two binary numbers. For example, let $A = 1011_2$, let $B = 1100_2$, and let $C = A +_2 B$. Then, the 4-bit addition yields C=0111 with a carry-out bit 1. This carry-out is equivalent to $B + \bar{B} + 1$ or $A + \bar{A} + 1$. Now, we compute $A = C -_2 B$. If $C \geq B$, then the ordinary binary subtraction $C - B$ yields the correct result of A. But if $C < B$, meaning there was a carry-out which was omitted during the addition, then the carry-out has to be added to $C - B$ to produce the correct result of A. We have $C - B + (B + \bar{B} + 1) = C + \bar{B} + 1$, or simply

$$C -_2 B = \begin{cases} C - B, & \text{if } C \geq B \\ C + \bar{B} + 1, & \text{if } C < B. \end{cases}$$

This equation also holds when A and B are n-bit binary numbers.

B. THE INITIALIZATION PHASE

In the initialization phase, system parameters are produced by the trusted server (i.e. the TTP). First of all, the trusted server chooses two large prime numbers q and n and a finite field F_q over $q > 2^w$, where w is the key size chosen. Then, the server

(1) specifies an elliptic curve equation $E: y^2 \equiv x^3 + ax + b \pmod q$ with the order n over F_q where $a, b \in F_q$ and $4a^3 + 27b^3 \neq 0 \pmod q$;
(2) selects a public point P and produces a cyclic additive group G by using P, in which both P and G are of the same order of n over E;
(3) selects a secure hash function $H_1 : U^7 \times G^2 \to Z_n^*$;
(4) publishes the system parameters, i.e., $\{G, P, H_1\}$.

Assume that user A and user B wish to join the system.

(1) User A (B) chooses his/her own password $pw_A(pw_B)$ from the password space D;
(2) The trusted server then derives the password key k_{PWA} (k_{PWB}) from pw_A (pw_B) for user A (user B).

C. THE AUTHENTICATION AND KEY EXCHANGE PHASE

Under the trusted server's help, users A and B are able to individually generate the same session keys for the following communication by using the following procedure.

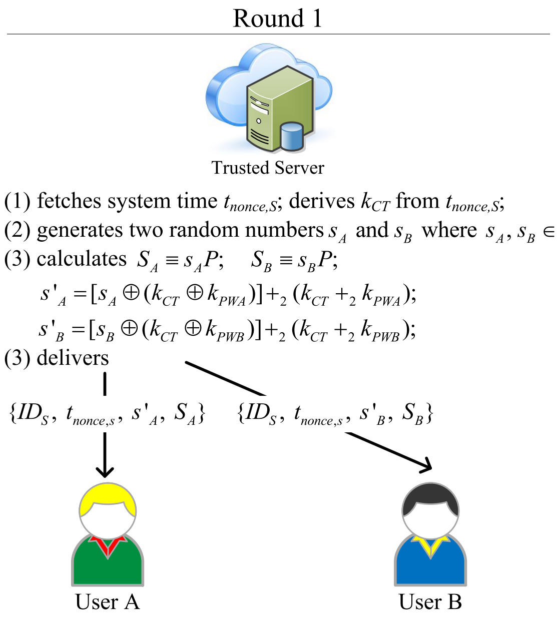

Round 1
Trusted Server
(1) fetches system time $t_{nonce,S}$; derives k_{CT} from $t_{nonce,S}$;
(2) generates two random numbers s_A and s_B where $s_A, s_B \in Z_n^*$;
(3) calculates $S_A \equiv s_AP$; $S_B \equiv s_BP$;
$s'_A = [s_A \oplus (k_{CT} \oplus k_{PWA})] +_2 (k_{CT} +_2 k_{PWA})$;
$s'_B = [s_B \oplus (k_{CT} \oplus k_{PWB})] +_2 (k_{CT} +_2 k_{PWB})$;
(3) delivers
$\{ID_S, t_{nonce,s}, s'_A, S_A\}$
$\{ID_S, t_{nonce,s}, s'_B, S_B\}$
User A
User B

FIGURE 1. Round 1 of the authentication and key exchange phase of the THMEP.

Round 1:

As shown in Fig. 1, the trusted server

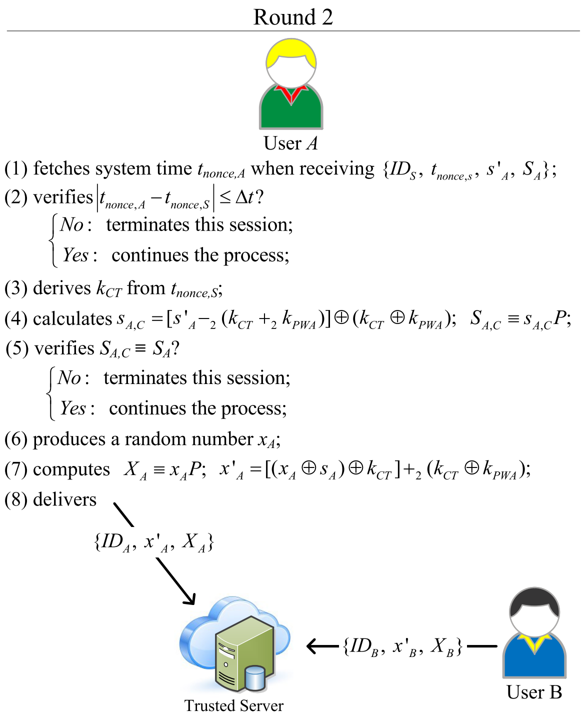
(1) fetches its system time $t_{nonce,S}$ and derives the current time encryption key k_{CT} from $t_{nonce,S}$ by using a hashing function H_2, i.e., $k_{CT} = H_2(t_{nonce,s})$;
(2) generates two random numbers s_A and s_B where $s_A, s_B \in Z_n^*$;
(3) calculates $S_A \equiv s_AP$; $S_B \equiv s_BP$; $s'_A = [s_A \oplus (k_{CT} \oplus k_{PWA})] +_2 (k_{CT} +_2 k_{PWA})$; $s'_B = [s_B \oplus (k_{CT} \oplus k_{PWB})] +_2 (k_{CT} +_2 k_{PWB})$;
(4) delivers $\{ID_S, t_{nonce,S}, s'_A, S_A\}$ to A and $\{ID_S, t_{nonce,S}, s'_B, S_B\}$ to B.

Round 2:

As shown in Fig. 2, on receiving the message sent by S, user A

(1) fetches its system time $t_{nonce,A}$;
(2) verifies whether or not $t_{nonce,A}$ satisfying $|t_{nonce,A} - t_{nonce,S}| \leq \Delta t$, in which Δt is a predefined time threshold for the allowable maximum transmission delay from the trusted server to user A. If not, the trusted server terminates this session. Otherwise, it
(3) derives k_{CT} from $t_{nonce,S}$;
(4) calculates $s_{A,C} = (s'_A -_2 (k_{CT} +_2 k_{PWA})) \oplus (k_{CT} \oplus k_{PWA})$ and $S_{A,C} \equiv s_{A,C} \cdot P$ where the subscript C indicates the value is calculated by user A;
(5) verifies whether or not $S_{A,C} \equiv S_A$; If not, it terminates this session. Otherwise, it
(6) produces a random number x_A;
(7) computes $X_A \equiv x_AP$ and $x'_A = [(x_A \oplus s_A) \oplus k_{CT}] +_2 (k_{CT} \oplus k_{PWA})$;
(8) delivers $\{ID_A, x'_A, X_A\}$ to the trusted server.

The same steps have been done by user B with the subscript


Round 2

User A

(1) fetches system time $t_{nonce,A}$ when receiving $\{ID_S, t_{nonce,s}, s'_A, S_A\}$;

(2) verifies $|t_{nonce,A} - t_{nonce,S}| \le \Delta t$?

$$\begin{cases} No: & \text{terminates this session;} \\ Yes: & \text{continues the process;} \end{cases}$$

(3) derives k_{CT} from $t_{nonce,S}$;

(4) calculates $s_{A,C} = [s'_A -_2 (k_{CT} +_2 k_{PWA})] \oplus (k_{CT} \oplus k_{PWA})$; $S_{A,C} \equiv s_{A,C}P$;

(5) verifies $S_{A,C} \equiv S_A$?

$$\begin{cases} No: & \text{terminates this session;} \\ Yes: & \text{continues the process;} \end{cases}$$

(6) produces a random number x_A;

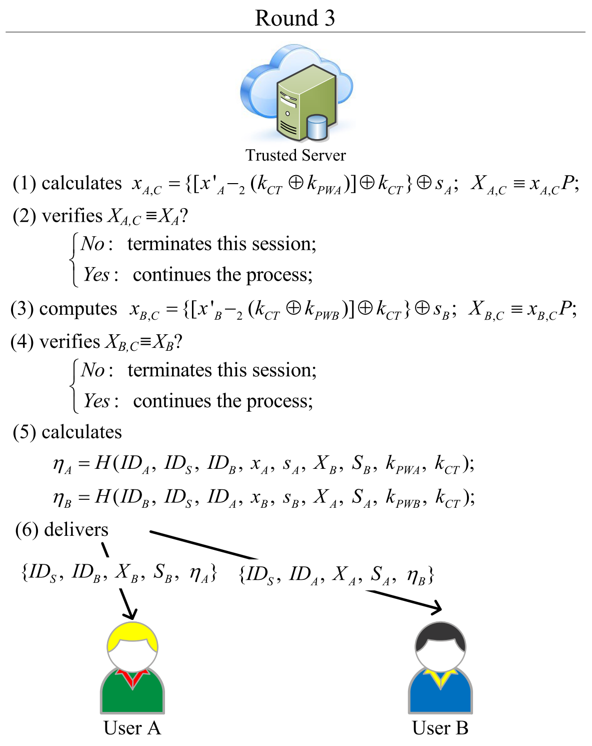
(7) computes $X_A \equiv x_A P$; $x'_A = [(x_A \oplus s_A) \oplus k_{CT}] +_2 (k_{CT} \oplus k_{PWA})$;

(8) delivers

$\{ID_A, x'_A, X_A\}$

$\{ID_B, x'_B, X_B\}$

FIGURE 2. Round 2 of the authentication and key exchange phase of the THMEP.


Round 3

Trusted Server

(1) calculates $x_{A,C} = \{[x'_A -_2 (k_{CT} \oplus k_{PWA})] \oplus k_{CT}\} \oplus s_A$; $X_{A,C} \equiv x_{A,C}P$;

(2) verifies $X_{A,C} \equiv X_A$?

$$\begin{cases} No: & \text{terminates this session;} \\ Yes: & \text{continues the process;} \end{cases}$$

(3) computes $x_{B,C} = \{[x'_B -_2 (k_{CT} \oplus k_{PWB})] \oplus k_{CT}\} \oplus s_B$; $X_{B,C} \equiv x_{B,C}P$;

(4) verifies $X_{B,C} \equiv X_B$?

$$\begin{cases} No: & \text{terminates this session;} \\ Yes: & \text{continues the process;} \end{cases}$$

(5) calculates

$\eta_A = H(ID_A, ID_S, ID_B, x_A, s_A, X_B, S_B, k_{PWA}, k_{CT})$;

$\eta_B = H(ID_B, ID_S, ID_A, x_B, s_B, X_A, S_A, k_{PWB}, k_{CT})$;

(6) delivers

$\{ID_S, ID_B, X_B, S_B, \eta_A\}$ $\{ID_S, ID_A, X_A, S_A, \eta_B\}$

FIGURE 3. Round 3 of the authentication and key exchange phase of the THMEP.

A substituted by B. Let's omit this redundant portion from this paper.

Round 3:

As shown in Fig. 3, when S receives message $\{ID_A, x'_A, X_A\}$ from user A, and $\{ID_B, x'_B, X_B\}$ from user B, it

(1) calculates $x_{A,C} = \{[x'_A -_2 (k_{CT} \oplus k_{PWA})] \oplus k_{CT} \oplus s_A$ and $X_{A,C} \equiv x_{A,C} \cdot P$;
(2) verifies whether or not $X_{A,C} \equiv X_A$; If not, S terminates this session. Otherwise, it
(3) computes $x_{B,C}$ and $X_{B,C}$ with similar equations shown in Step (1);
(4) verifies whether or not $X_{B,C} \equiv X_B$; If not, S terminates this session. Otherwise, it
(5) calculates $\eta_A = H(ID_A, ID_S, ID_B, x_A, s_A, X_B, S_B, k_{PWA}, k_{CT})$, and $\eta_B = H(ID_B, ID_S, ID_A, x_B, s_B, X_A, S_A, k_{PWB}, k_{CT})$;
(6) delivers $\{ID_S, ID_B, X_B, S_B, \eta_A\}$ to A and $\{ID_S, ID_A, X_A, S_A, \eta_B\}$ to B.

Round 4:

As shown in Fig. 4, upon receiving message $\{ID_S, ID_B, X_B, S_B, \eta_A\}$ from the trusted server, user A

(1) computes $\eta_{A,C} = H(ID_A, ID_S, ID_B, x_A, s_A, X_B, S_B, k_{PWA}, k_{CT})$;
(2) checks to see whether or not $\eta_{A,C} = \eta_A$; If not, it terminates this session. Otherwise, it
(3) computes $K_0 \equiv x_A X_B$; $K_1 \equiv x_A S_B$; $K_2 \equiv s_A X_B$; and $K_3 \equiv s_A S_B$;
(4) generates $SK_t = (K_i +_2 K_j) \oplus K_k$, where $1 \le i \le j \le 4$, $1 \le k \le 4$, $1 \le t \le 40$, and

$$t = \begin{cases} 0 + 4(j - i) + k, & \text{if } i = 1 \\ 16 + 4(j - i) + k, & \text{if } i = 2 \\ 28 + 4(j - i) + k, & \text{if } i = 3 \\ 36 + 4(j - i) + k, & \text{if } i = 4. \end{cases}$$

Analogously, user B

(1) computes $\eta_{B,C} = H(ID_B, ID_S, ID_A, x_B, s_B, X_A, S_A, k_{PWB}, k_{CT})$;
(2) checks to see whether or not $\eta_{A,C} = \eta_A$; If not, it terminates this session. Otherwise, it
(3) computes $K_0 \equiv x_B X_A$; $K_1 \equiv s_B X_A$; $K_2 \equiv x_B S_A$; and $K_3 \equiv s_B S_A$;
(4) generates $SK_t = (K_i +_2 K_j) \oplus K_k$, where $1 \le i \le j \le 4$, $1 \le k \le 4$, $1 \le t \le 40$, and

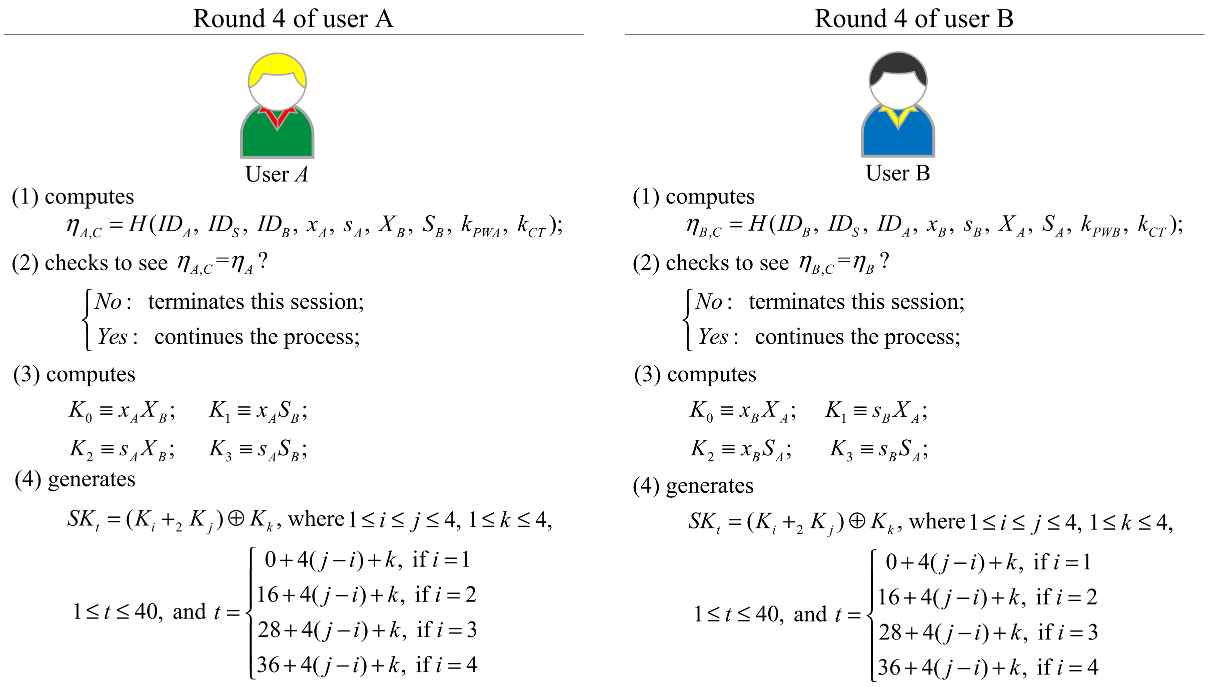
$$t = \begin{cases} 0 + 4(j - i) + k, & \text{if } i = 1 \\ 16 + 4(j - i) + k, & \text{if } i = 2 \\ 28 + 4(j - i) + k, & \text{if } i = 3 \\ 36 + 4(j - i) + k, & \text{if } i = 4. \end{cases}$$

V. SECURITY ANALYSIS

In this section, the security of the THMEP is evaluated.

A. SECURITY ANALYSIS

Let X and Y be two keys, each of which is m bits in length. According to the proofs proposed by Huang *et al.* [32], [33], the probability p with which to recover the value of (X, Y) from illegally intercepted $X \oplus Y$ on one trial is $p = 1/2^m$. The recovering probability of $X +_2 Y$ is also $1/2^m$. By using these two fundamental concepts, probability p with which to



Round 4 of user A

User A

(1) computes

$$\eta_{A,C} = H(ID_A,\ ID_S,\ ID_B,\ x_A,\ s_A,\ X_B,\ S_B,\ k_{PWA},\ k_{CT});$$

(2) checks to see $\eta_{A,C} = \eta_A$?

$$\begin{cases} No: & \text{terminates this session;} \\ Yes: & \text{continues the process;} \end{cases}$$

(3) computes

$$K_0 \equiv x_A X_B; \quad K_1 \equiv x_A S_B;$$
$$K_2 \equiv s_A X_B; \quad K_3 \equiv s_A S_B;$$

(4) generates

$$SK_t = (K_i +_2 K_j) \oplus K_k, \text{ where } 1 \le i \le j \le 4,\ 1 \le k \le 4,$$

$$1 \le t \le 40, \text{ and } t = \begin{cases} 0 + 4(j-i) + k, & \text{if } i = 1 \\ 16 + 4(j-i) + k, & \text{if } i = 2 \\ 28 + 4(j-i) + k, & \text{if } i = 3 \\ 36 + 4(j-i) + k, & \text{if } i = 4 \end{cases}$$

Round 4 of user B

User B

(1) computes

$$\eta_{B,C} = H(ID_B,\ ID_S,\ ID_A,\ x_B,\ s_B,\ X_A,\ S_A,\ k_{PWB},\ k_{CT});$$

(2) checks to see $\eta_{B,C} = \eta_B$?

$$\begin{cases} No: & \text{terminates this session;} \\ Yes: & \text{continues the process;} \end{cases}$$

(3) computes

$$K_0 \equiv x_B X_A; \quad K_1 \equiv s_B X_A;$$
$$K_2 \equiv x_B S_A; \quad K_3 \equiv s_B S_A;$$

(4) generates

$$SK_t = (K_i +_2 K_j) \oplus K_k, \text{ where } 1 \le i \le j \le 4,\ 1 \le k \le 4,$$

$$1 \le t \le 40, \text{ and } t = \begin{cases} 0 + 4(j-i) + k, & \text{if } i = 1 \\ 16 + 4(j-i) + k, & \text{if } i = 2 \\ 28 + 4(j-i) + k, & \text{if } i = 3 \\ 36 + 4(j-i) + k, & \text{if } i = 4 \end{cases}$$

FIGURE 4. Round 4 of the authentication and key exchange phase of the THMEP.

recover the value of s_A from known s'_A is $p = 1/2^m$. We will show this in Theorem 1.

Theorem 1: Assume that both random parameter s_A and communication key s'_A are m bits in length. The probability p with which to recover the value of s_A from known s'_A is $p = 1/2^m$.

Proof: According to previous description,

$$\begin{aligned} s'_A &= [s_A \oplus (k_{CT} \oplus k_{pwA})] +_2 (k_{CT} +_2 k_{pwA}) \\ &= (s_A \oplus k_1) +_2 k_2 \end{aligned} \qquad (1)$$

where $k_1 = k_{CT} \oplus k_{pwA}$ and $k_2 = k_{CT} +_2 k_{pwA}$. If s'_A is known to hackers and utilized to recover s_A, then k_1 and k_2 must be obtained beforehand. However, k_1 and k_2 are time-variable, i.e., different $t_{nonce,S}$s will result in different k_1s and k_2s, which are computed by the trusted server, meaning that k_1 and k_2 provide higher variability and security than only using k_{pwA} does. Besides, different s'_As are generated by invoking different s_As and k_{CT}s. Hence, the collection of a large number of s'_A does not work in cracking the connection keys and recovering s_A. Then, due to invoking two-dimensional operation (i.e., $+_2$ and $\oplus$), the probability p with which to recover s_A, k_1, and k_2 from s'_A by using Eq. (1) is $(1/2^m)^2$ (i.e., one $\oplus$ and one $+_2$, $1/2^m \times 1/2^m$) which is much smaller than $1/2^m$ where $1/2^m$ is the probability with which to blind guess the value of s_A on one trial when s'_A is known. It indicates that, no matter whether Eq. (1) is employed or not, the probability p with which to recover the value of s_A from a known s'_A is $p = 1/2^m$. Q.E.D.

Theorem 1 can also be used when recovering the values of s_B, x_A, and x_B. In the THMEP, the important parameters s_A, s_B, x_A, and x_B are well-protected by k_{CT}, K_{PWA} and K_{PWB}, implying that the security of the THMEP is higher than that of the 3MPAKE. Theorem 2 shows the details.

Theorem 2: In the THMEP, replay attacks can be effectively defended by using t_{nonce}, s'_A and S_A.

Proof: If hackers invalidly duplicate the message $\{ID_S, t_{nonce,S}, s'_A, S_A\}$ and resend it to user A, then $t_{nonce,S}$ contained in this message is not current time so that $|t_{nonce,A} - t_{nonce,S}| > \Delta t$ where Δt is a predefined short time period. The message will be discarded by user A. If hackers modify $t_{nonce,S}$ to satisfy the limitation of Δt, and resend the message to user A, then user A will generate another k_{CT} in Step (3) of Round 2, and obtain a wrong $s_{A,C}$ in Step (4) of Round 2. Consequently, the verification of the equality of $S_{A,C}$ and S_A in Step (5) will fail. Since hackers do not know k_{PWA}, they cannot generate valid s'_A and S_A by using current $t_{nonce,S}$, showing that in the THMEP, $t_{nonce,S}$, s'_A and S_A together can effectively defend the replay attacks. Q.E.D.

Theorem 3 will show the authentication and nonrepudiation features of the THMEP.

Theorem 3: In Round 3 of the authentication and key exchange phase, $\eta_A = H(ID_A,\ ID_S,\ ID_B,\ x_A,\ s_A,\ X_B,\ S_B,\ k_{pwA},\ k_{CT})$ is a code with authentication and nonrepudiation features.

Proof:

(Proof of Authentication): To correctly generate the value of hash function $H(ID_A, ID_S, ID_B, x_A, s_A, X_B, S_B, k_{pwA}, k_{CT})$, the following two steps are required:

(1) The trusted server decrypts x'_A to obtain x_A by using s_A , k_{CT} and k_{pwA} (see Step (7) of Round 2).

(2) The trusted server checks to see whether or not $X_{A,C} = X_A$ in Step (2) of Round 3.

These two steps imply that only the hackers who have acquired the parameters s_A and k_{pwA} can correctly generate the values of η_A in Step (1) of Round 4 for user A. Hence, only the legitimate user who has parameters x_A, s_A, and k_{pwA} can generate the correct η_A, i.e., $\eta_{A,C} = \eta_A$ (see Step (2) of Round 4). Those illegitimate hackers who have no correct parameters x_A, s_A, and k_{pwA} cannot achieve this.

User B has the similar phenomena and description. But let's omit them here.

(Proof of Nonrepudiation): From the analysis above, only the legitimate user who knows the parameters x_A, s_A, k_{CT} and k_{pwA} can make $\eta_{A,C} = H(ID_A, ID_S, ID_B, x_A, s_A, X_B, S_B, k_{pwA}, k_{CT}) = \eta_A$ (i.e., Step (2) of Round 4). This shows that the message is sent by the one who has those valid parameters, indicating that the user is a legitimate one. Q.E.D.

TABLE 2. Security comparisons among Chen et al. [27], Yang et al. [28], the 3MPAKE [11], and the THMEP.

Security attribution	Chen et al. [29]	Yang et al. [30]	3MPAKE [11]	THMEP
Mutual Authentication	✓ (partial)	✓ (partial)	✓ (partial)	✓ (full)
Replay attack	×	×	×	✓
Eavesdropping attack	×	×	×	✓
Known-key attack	✓	✓	✓	✓
Impersonation attack	✓	×	✓	✓
Forgery attack	×	×	×	✓

B. SECURITY COMPARISON

The THMEP protocol provides mutual authentication and is secure against five popular attacks. Table 2 compares the security among the THMEP and other related protocols, including Chen et al. [29], Yang et al. [30] and the 3MPAKE [11].

1) FULL MUTUAL AUTHENTICATION

In the THMEP, each round of the authentication and key exchange phase has its own authentication mechanism except the first round. In Step (5) of Round 2, user A verifies the trusted server's message by checking the equality of $S_{A,C}$ and S_A. Only the legitimate server who knows k_{PWA} can pass the verification. In Step (2) of Round 3, the trusted server verifies user A by checking the equality of $X_{A,C}$ and X_A. Only legitimate user A who owns correct k_{PWA} can correctly decrypt s'_A to obtain s_A and then generate valid X_A to pass the verification. In Step (4) of Round 3, S verifies user B with the similar method. In Step (2) of Round 4, user A verifies the trusted server by checking to see whether or not $\eta_{A,C} = \eta_A$. Only legitimate server who has the parameters s_A, x_A and k_{PWA} can correctly generate valid η_A, and pass the verification. In Step (4) of Round 4, user B does the same. Hence, the THMEP is a full mutual authentication protocol. However, the 3MPAKE [11] and the two schemes in [29] and [30] lack an authentication mechanism between user A and the server in their Round 2s, which may cause forgery attack and decrease the efficiency of finding invalid users. Namely, the 3MPAKE, Chen's scheme [29], and Yang's system [30] only individually provide a partial mutual authentication.

2) REPLAY ATTACK

Assume the trusted server would like to send messages to user A, and a hacker Z intercepts the messages and disguises himself/herself as the trusted server to transmit them to user A. In this case, user A in the 3MPAKE can successfully authenticate Z by its key exchange procedure, meaning that there is no mechanism to prevent this protocol from the replay attack. In the THMEP, Theorem 2 shows that it can defend the attack effectively.

3) EAVESDROPPING ATTACK

When Z captures a large number of messages from the underlying network, he/she may be able to obtain some important parameters, such as user's password. In the 3MAPKE, some keys are fixed and static, for example, the verifier v_A is derived from user A's password. If user A does not change the password, the verifier v_A will remain unchanged. Hence, it is relatively easier for Z to extract important parameters from captured messages. In the THMEP, the password keys k_{PWA} and k_{PWB} are encrypted by time key k_{CT} (see Step (3) of Round 1) which is dynamic for different sessions established at different time points. Thus the parameters generated to protect delivered messages and keys in different sessions also vary. Even though Z has captured a large amount of messages from the network, he/she is still unable to extract users' keys from the captured messages. Hence, the THMEP is able to thwart the eavesdropping attack.

4) KNOWN-KEY ATTACK

This type of attack occurs when a hacker knows the key. He/she can find the ciphering mechanism, resulting in the fact that the keys generated later may be exposed. In the THMEP, as mentioned above, the session keys established for a session are quite different from those generated in other sessions. If Z can obtain one of the previous session keys, he/she still does not know those time keys and random numbers used to generate those session keys for later sessions. Thus, the THMEP can prevent the known-key attack effectively.

5) IMPERSONATION ATTACK

When Z wants to impersonate user A, he/she intercepts the message sent by the trusted server to user A in Round 1, and guesses s_A from the captured message. However, due to the lack of correct k_{PWA}, according to Theorem 1, Z cannot correctly decrypt s'_A to obtain s_A, and hence cannot correctly generate x'_A since he/she does not have parameters s_A

TABLE 3. The times consumed by seven operations (unit: ms).

Operations	Symbol	Key length (bits)			
		160	256	512	1024
Elliptic curve point multiplication	T_{em}	236.63	243.74	282.92	377.51
Hash function operation	T_h	0.41	0.42	2.45	8.29
Binary addition	T_{ba}	0.06	0.07	0.09	0.13
Binary multiplication	T_{om}	0.06	0.07	0.15	0.26
Pseudo-random function operation	T_f	4.69	4.79	4.85	5.38
Inverse operation	T_{inv}	0.82	0.88	1.54	2.36
Logical XOR operation	T_{xor}	0.03	0.03	0.06	0.10

TABLE 4. Computational efforts of the 3MPAKE and THMEP.

Computational cost	3MPAKE [11]	THMEP
User A	$10T_{em}+2T_h+5T_{ba}+2T_{om}+4T_f+T_{inv}$	$6T_{em}+1T_h+16T_{ba}+18T_{xor}$
User B	$10T_{em}+2T_h+5T_{ba}+2T_{om}+4T_f+T_{inv}$	$6T_{em}+1T_h+16T_{ba}+18T_{xor}$
The trusted server	$28T_{em}+4T_h+2T_{inv}$	$4T_{em}+2T_h+6T_{ba}+10T_{xor}$
Total	$48T_{em}+8T_h+10T_{ba}+4T_{om}+8T_f+4T_{inv}$	$16T_{em}+4T_h+38T_{ba}+46T_{xor}$

T_{em}: the time required to perform an elliptic curve point multiplication;
T_h: the time required to calculate a hash function operation;
T_{ba}: the time required to compute a binary addition;
T_{om}: the time required to compute a binary multiplication;
T_f: the time required to accomplish a pseudo-random function operation;
T_{inv}: the time required to calculate an inverse operation;
T_{xor}: the time required to finish a logical XOR operation;

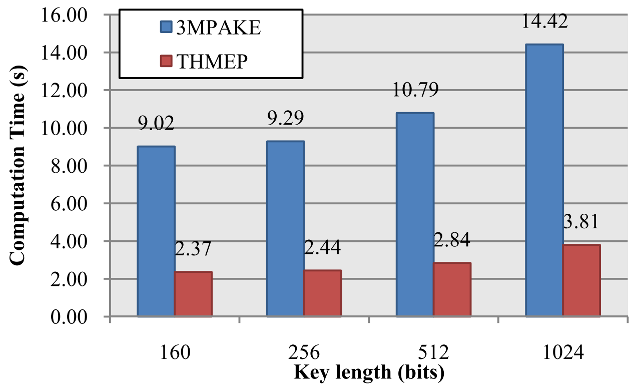


FIGURE 5. The computation times of THMEP and 3MPAKE.

TABLE 5. Performance improvement achieved by the THMEP over the 3MPAKE on different key lengths.

Key length (bits)			
160	256	512	1024
3.81	3.81	3.80	3.78

and k_{PWA}. Thus, Z is unable to pass the verification performed in Step (2) of Round 3, showing that the THMEP can effectively defend the impersonation attack.

6) FORGERY ATTACK

In Round 1, Z may pretend himself/herself as the trusted server by issuing a valid S_A and a valid s'_A (see Step (3) of Round 1) which are invalidly captured. However, the captured s'_A is encrypted by k_{CT} which is the current time encryption key of the trusted server, i.e., k_{CT} is derived from $t_{nonce,S}$ (see Step (1) of Round 1). If Z sends a fake $t_{nonce,S}$ in Step (3) of Round 1, then a wrong s_A will be generated in Step (4) of Round 2 and cannot pass the verification in Step (5). Furthermore, Z may generate a fake message which is sent to S in Step (3) of Round 1. However, he/she does not know the correct values of K_{PWA}. Therefore, the correct value of s_A cannot be generated in Step (4) of Round 2. In other words, the message cannot pass the verification process, meaning that the THMEP can effectively defend the Forgery attack.

VI. PERFORMANCE ANALYSIS

Table 3 summarizes the time consumed by seven operations (as shown) on key sizes of 160 bits, 256 bits, 512 bits, and 1024 bits. The simulation is performed on a notebook computer with 1.6GHz Intel Atom CPU, 1GB RAM, Windows XP operating system, and C++ compiler. It is clear that an ECC point multiplication consumes the longest computation time compared with those consumed by other operations. That means the ECC multiplication is the dominant factor in the total computation time and contributes most of the computational cost. Note that in the 3MPAKE, the server undergoes 48 ECC multiplications.

The performance comparison between the 3MPAKE and the THMEP is presented in Table 4 when four session keys are generated. We assume that the size of P, the output size of hash function, and the size of user's identity are 160, 160, and 32 bits, respectively. The computational costs for different parties (user A, user B and S) are shown in this table. Although the logic XOR operations and binary additions are more frequently used in the THMEP, the number of ECC multiplications employed is reduced from 48 to 16.

Fig. 5 depicts the total computation time of the 3MPAKE and the THMEP. As the key length increases, the total computation time of the THMEP grows slower than that of 3MPAKE. In average, the processing speed of the THMEP is 3.78 (=14.42/3.81, taking 1024 bits shown in Fig. 5 as an example) times that of the 3MPAKE. Table 5 shows the performance improvement (PI) achieved by the THMEP over the 3MPAKE on different key lengths where PI is defined as the time consumed by the 3MPAKE over the time spent by the THMEP on a specific key length.

$$PI_{Key\ length} = \frac{the\ time\ consumed\ by\ the\ 3MPAKE}{the\ time\ spent\ by\ the\ THMEP}$$

Overall, our protocol utilizes less ECC multiplications and saves a significant amount of computation time. In the 3MPAKE, the relatively high computation burden of the trusted server has lengthened its communication latency and then the response time. Hence, the THMEP can keep higher performance improvement even when the key length is longer, that is, the THMEP provides higher security level and better scalability than the 3MPAKE does when their computation times are the same.

VII. CONCLUSION AND FUTURE STUDIES

In this paper, we propose the THMEP to enhance the computational performance and security level of the key exchange process in mobile communication. Comparing with previous studies, the THMEP has lower computation and communication costs and provides 40 session keys. The processing speed of the THMEP is 3.78 times faster than that of the 3MPAKE when the key length is 1024 bits. Besides, the THMEP is reliable since it can resist replay, eavesdropping, known-key, impersonation, and forgery attacks, and offer full mutual authentication. In other words, the THMEP has the features of high efficiency, scalability, reliability, and throughput, and is in particular suitable for mobile applications.

In the THMEP, the important parameters s_A, s_B, x_A, and x_B are encrypted by the two-dimensional operation, time key and users' passwords. Although the computation costs have been greatly reduced, as shown in Table 4, a total of 16 ECC multiplication is still required. They mainly come from Round 4 of the authentication and key exchange phase. Thus, how to reduce the number of ECC point multiplication is an important issue for further improving the THMEP's performance. In addition, the energy limitation of mobile devices is also a key factor in developing an adaptive THMEP for heterogeneous environments. We would also like to derive the reliability model, energy model and behaviour model for the THMEP so that users can know its reliability, energy consumption and behaviours before using it. These constitute our future studies.

REFERENCES

[1] U. Varshney and R. Vetter, "Mobile commerce: Framework, applications and networking support," *J. Mobile Netw. Appl.*, vol. 7, no. 3, pp. 185–198, Jun. 2002.

[2] E. W. T. Ngai and A. Gunasekaran, "A review for mobile commerce research and applications," *Decision Support Syst.*, vol. 43, no. 1, pp. 3–15, Feb. 2007.

[3] E. Turban, J. K. Lee, D. King, T. P. Liang, and D. Turban, *Electronic Commerce*, Upper Saddle River, NJ, USA: Prentice-Hall, 2010.

[4] J. B. Schafer, J. Konstan, and J. Riedl, "Recommender systems in E-commerce," in *Proc. ACM Conf. Electron. Commerce*, Nov. 1999, pp. 158–166.

[5] L. M. Adleman, R. L. Rivest, and A. Shamir, "Cryptographic communications system and method," U.S. Patent 4 405 829, Sep. 20, 1983.

[6] Y. L. Huang and F. Y. Leu, "Constructing a secure point-to-point wireless environment by integrating Diffie–Hellman PKDS RSA and stream ciphering for user known to each other," *J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl.*, vol. 2, no. 3, pp. 96–107, Sep. 2011.

[7] Y. L. Huang, F. Y. Leu, I. You, Y. K. Sun, and C. C. Chu, "A Secure wireless communication system integrating RSA, Diffie–Hellman PKDS, intelligent protection-key chains and a data connection core in a 4G environment," *J. Supercomput.*, vol. 67, no. 3, pp. 635–652, Mar. 2014.

[8] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[9] S. Chakraborty, S. Raghuraman, and C. Pandu, "A pairing-free, one round identity based authenticated key exchange protocol secure against memory-scrapers," *J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl.*, vol. 7, no. 1, pp. 1–22, Mar. 2016.

[10] D. Yang and B. Yang, "A novel multi-factor authenticated key exchange scheme with privacy preserving," *J. Internet Services Inf. Secur.*, vol. 1, nos. 2–3, pp. 44–56, Aug. 2011.

[11] W. M. Li, Q. Y. Wen, Q. Su, H. Zhang, and Z. P. Jin, "Password-authenticated multiple key exchange protocol for mobile applications," *China Commun.*, vol. 9, no. 1, pp. 64–72, Jan. 2012.

[12] T. Pandit, R. Barua, and S. Tripathy, "eCK secure single round ID-based authenticated key exchange protocols with master perfect forward secrecy," *J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl.*, vol. 5, no. 4, pp. 65–85, Dec. 2014.

[13] K. L. Tsai, F. Y. Leu, and J. S. Tan, "An ECC-based secure EMR transmission system with data leakage prevention scheme," *Int. J. Comput. Math.*, vol. 93, no. 2, pp. 367–383, Feb. 2016.

[14] C. C. Lee, C. T. Li, and C. W. Hsu, "A three-party password-based authenticated key exchange protocol with user anonymity using extended chaotic maps," *Nonlinear Dyn.*, vol. 73, no. 1, pp. 125–132, Jul. 2013.

[15] S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," in *Proc. Symp. Secur. Privacy*, May 1992, pp. 72–84.

[16] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn. Bruges*, May 2000, pp. 139–155.

[17] M. Abdalla and D. Pointcheval, "Simple password-based encrypted key exchange protocols," in *Topics Cryptology*. Berlin, Germany: Springer-Verlag, 2005, pp. 191–208.

[18] A.-F. Sui *et al.*, "An improved authenticated key agreement protocol with perfect forward secrecy for wireless mobile communication," in *Proc. IEEE Wireless Commun. Netw. Conf.*, vol. 4. Mar. 2005, pp. 2088–2093.

[19] J. W. Lo, C. C. Lee, M. S. Hwang, and Y. P. Chu, "A secure and efficient ECC-based AKA protocol for wireless mobile communications," *Int. J. Innov. Comput., Inf. Control*, vol. 6, no. 11, pp. 5249–5258, Nov. 2010.

[20] M. Abdalla, P. A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proc. Int. Workshop Theory Pract. Public Key Cryptogr.*, Jan. 2005, pp. 65–84.

[21] R. X. Lu and Z. F. Cao, "Simple three-party key exchange protocol," *Comput. Secur.*, vol. 26, no. 1, pp. 94–97, Feb. 2007.

[22] H. R. Chung and W. C. Ku, "Three weaknesses in a simple three-party key exchange protocol," *Inf. Sci.*, vol. 178, no. 1, pp. 220–229, Jan. 2008.

[23] H. Guo, Z. J. Li, Y. Mu, and X. Y. Zhang, "Cryptanalysis of Simple Three-party Key Exchange Protocol," *Comput. Security*, vol. 27, nos. 1–2, pp. 16–21, Mar. 2008.

[24] T. Y. Chang, M. S. Hwang, and W. P. Yang, "A communication-efficient three-party password authenticated key exchange protocol," *Inf. Sci.*, vol. 181, no. 1, pp. 217–226, Jan. 2011.

[25] E. J. Yoon and K. Y. Yoo, "Cryptanalysis of a simple three-party password-based key exchange protocol," *Int. J. Commun. Syst.*, vol. 24, no. 4, pp. 532–542, Apr. 2011.

[26] W. S. Robert, *Internet Security Glossary*, document 55, May 2000.

[27] B. W. Simon and M. Alfred, "Unknown key-share attacks on the station-to-station (STS) protocol," in *Proc. Int. Workshop Pract. Theory Public Key Cryptogr.*, Mar. 1999, pp. 154–170.

[28] D. Yun and H. Patrick, "Undetectable on-line password guessing attacks," *ACM SIGOPS Oper. Syst. Rev.*, vol. 29, no. 4, pp. 77–86, Oct. 1995.

[29] T. H. Chen, W. B. Lee, and H. B. Chen, "A round- and computation-efficient three-party authenticated key exchange protocol," *J. Syst. Softw.*, vol. 81, no. 9, pp. 1581–1590, Sep. 2008.

[30] J. H. Yang and C. C. Chang, "An efficient three-party authenticated key exchange protocol using elliptic curve cryptography for mobile commerce environments," *J. Syst. Softw.*, vol. 82, no. 9, pp. 1497–1502, Sep. 2009.

[31] D. R. Stinson, *Cryptography: Theory and Practice*, 3rd ed. London, U.K.: Chapman & Hall, 2006.

[32] Y. F. Huang, F. Y. Leu, C. H. Chiu, and I. L. Lin, "Improving security levels of IEEE 802.16e authentication by involving Diffie–Hellman PKDS," *J. Universal Comput. Sci.*, vol. 17, no. 6, pp. 891–911, 2010.

[33] Y. L. Huang, F. Y. Leu, and K. C. Wei, "A secure communication over wireless environments by using a data connection core," *Math. Comput. Model.*, vol. 58, nos. 5–6, pp. 1459–1474, Sep. 2013.

KUN-LIN TSAI (M'00) received the B.S. degree in computer and information science from Tung-Hai University, Taichung, Taiwan, in 1999, and the M.S. degree in computer science and information engineering from National Taiwan University, Taipei, in 2001, and the Ph.D. degree in electrical engineering from National Taiwan University in 2006. He held a post-doctoral position with the National Taiwan University of Science and Technology in 2007. He is currently an Associate Professor and the Chairman with the Department of Electrical Engineering, Tunghai University. His research interests are low-power system design, information security system, and VLSI design.

YI-LI HUANG (M'13) received the master's degree from the National Central University of Physics, Taiwan, in 1983. He is currently an Associate Professor with Tunghai University, Taiwan, and the Director with the Information Security Laboratory. His research interests include security of network and wireless communication, solar active-tracking system, pseudorandom number generator design, and file protection theory.

FANG-YIE LEU (M'87) received the B.S., M.S., and Ph.D. degrees from the National Taiwan University of Science and Technology, Taiwan, in 1983, 1986 and 1991, respectively, and the M.S. degree from the Knowledge Systems Institute, USA, in 1990. He is currently a Workshop Organizer of CWECS and MCNCS workshops, a Professor with Tunghai University, Taiwan, the Director with the Database and Network Security Laboratory, and the Chairperson with the Department of Information Management, Tunghai University. His research interests include wireless communication, network security, grid applications and Chinese natural language processing. He is also a member of the IEEE Computer Society and one of the editorial board members of at least six international journals.

ILSUN YOU (SM'13) received the M.S. and Ph.D. degrees in computer science from Dankook University, Seoul, South Korea, in 1997 and 2002, respectively, and the Ph.D. degree from Kyushu University, Japan, in 2012. From 1997 to 2004, he was with the THIN multimedia Inc., Internet Security Co., Ltd. and Hanjo Engineering Co., Ltd. as a Research Engineer. He is currently an Associate Professor with the Department of Information Security Engineering, Soonchunhyang University. He is a fellow of the IET. He has served or is currently serving as the Organizer of international conferences and workshops, such as MobiWorld, MIST, SeCIHD, AsiaARES, IMIS, and so forth. He is the Editor-in-Chief of the *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*. He is on the Editorial Board of the *Information Sciences*, the *Journal of Network and Computer Applications*, the *International Journal of Ad Hoc and Ubiquitous Computing*, the *Intelligent Automation and Soft Computing*, the *Computing and Informatics*, the *Journal of High Speed Networks*, and so forth. His main research interests include internet security, authentication, access control, and formal security analysis.