IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 25, 2017, accepted November 22, 2017, date of publication November 27, 2017,
date of current version December 22, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2777840

Efficient and Secure Time-Key Based Single Sign-On Authentication for Mobile Devices

VANGA ODELU[1], ASHOK KUMAR DAS[2], (Member, IEEE), KIM-KWANG RAYMOND CHOO[3], (Senior Member, IEEE), NEERAJ KUMAR[4], (Senior Member, IEEE), AND YOUNGHO PARK[5], (Member, IEEE)

[1]Department of Computer Science and Engineering, Indian Institute of Information Technology Chittoor, Sricity 517 588, India
[2]Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India
[3]Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249 USA
[4]Department of Computer Science and Engineering, Thapar University, Patiala 147 004, India
[5]School of Electronics Engineering, Kyungpook National University, Daegu 41566, South Korea

Corresponding author: Youngho Park (parkyh@knu.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea, Ministry of Science, ICT, and Future Planning, under Grant 2017R1A2B1002147 and in part by the BK21 Plus Project through the Ministry of Education, South Korea, under Grant 21A20131600011.

ABSTRACT In recent years, mobile devices are becoming an integrated part of our society, and this reinforces the need for security and privacy without incurring additional communication and computation costs. In this paper, we propose a new efficient privacy preserving time-key-based single sign-on (TK-SSO) authenticated key management protocol for mobile devices using elliptic curve cryptography. This allows us to achieve the desirable security properties along with significantly reduced computation and communication costs. TK-SSO also supports the revocation of mobile users and servers. We prove the security of TK-SSO in a widely accepted adversary real-or-random model, as well as using Burrows–Abadi–Needham (BAN) logic and the Automated Validation of Internet Security Protocols and Applications (AVISPA) simulation tool to demonstrate that TK-SSO can resist various known attacks. We then evaluate the performance of TK-SSO and three related protocols to demonstrate its utility.

INDEX TERMS Key distribution, temporal key, credential privacy, session-key security, BAN logic, AVISPA simulation.

I. INTRODUCTION

With the popularity of mobile devices, such as Android and iOS devices, more traditional business transactions are being conducted online 24/7, regardless of their physical locations as long as they have Internet connections [1]–[4]. While this affords us many benefits, the openness nature of the wireless environment also exposes the users and system to a wide range of attacks [5], [6].

An example mobile cloud computing architecture is shown in Fig. 1. In this architecture, the mobile devices can access cloud services in two manners: either through mobile network (telecom network) or through access points. A mobile user can use the mobile devices to access different services from different service providers via a wireless local area network (WLAN) or 3G/4G telecommunication networks, which may or may not be secure. The distributed locations of the service providers make it convenient for subscribers to access various resources. However, the design of distributed authentication protocol to ensure secure communication while providing low computational overheads for these mobile users is essential. In recent years, a large number of authentication protocols for single-server environment have been proposed [7], [8]. To cater to the wide ranging and increasing demands for richer and interactive user services, more multi-server environments are deployed in practical applications. However, existing authentication protocols designed for single-server environment are not suited for such multi-server environment deployment because a user has to register in every single server and remembers all passwords for the many different servers. Single sign-on (SSO) authentication is a viable solution since it allows a user with a single credential to access multiple service providers; thus, this gives rise to multi-server authentication (MSA) protocol.

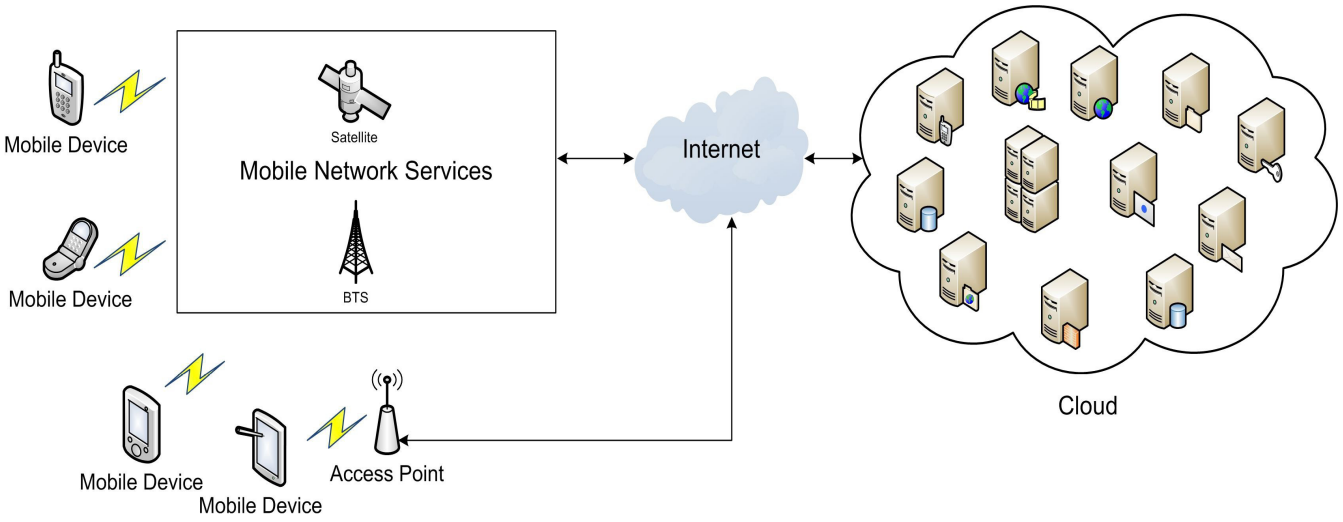

FIGURE 1. A mobile cloud architecture [1].

In conventional SSO schemes, OpenID has been widely adopted by many Internet service providers such as Yahoo and Google, with over 50,000 websites reportedly using OpenID as their authentication scheme. In OpenID, three-parties are involved in the authentication process. In addition, the OpenID specifications strongly recommend the use of SSL network connection for all message transmissions. Since SSL technique is based on RSA public key cryptosystem, SSL implementation requires heavy computation costs. Such costs may not be realistic for deployment on a mobile device (e.g. when service-requests from mobile users are considered) [9]–[12]. The 256-bit elliptic curve cryptosystem (ECC) provides equivalent level of security as in a 3072-bit RSA public key [13]. Thus, ECC is more efficient for mobile users as compared to RSA.

The typical requirements of a multi-server authentication (MSA) protocol are user anonymity, session key (SK)-security, and mutual authentication. Recent literature have also suggested the need for revocation and re-registration features to be incorporated in MSA protocols due to the very real risk of a static authentication token being lost/compromised and subsequently used by an adversary to impersonate the user [18], [28]. Although a number of MSA protocols have been proposed in the literature, most of these protocols do not support all the necessary security features. The design of a provably-secure and efficient MSA protocol supporting both revocation and un-traceability remains a research challenge [27], [29].

Existing MSA protocols can be broadly categorized into three-party based (Category 1; see [14]–[18]), and two-party based (Category 2; see [22]–[28]) – see Table 1. In Category 1, a registration center, a server, and a user are involved in the authentication process, and protocols in Category 2 only consist of a server S and a user U in the authentication process. In Category 1, no MSA protocol with the exception of Odelu *et al.* [18] provides all necessary security features. We also observe that the majority of the existing Category 2 MSA protocols fail to provide all necessary security features. This is the gap we seek to address in this paper, where we present an efficient privacy preserving time-key based single sign-on authenticated key management protocol (TK-SSO) for a multi-server environment supporting all necessary security features.

A. RELATED WORK

This section briefly reviews several multi-server authentication schemes proposed in the literature.

The design of MSA protocols has not been a smooth journey. For example, in 2013, Liao and Hsiao [23] proposed a pairing-based MSA protocol using self-certified public keys for mobile clients. However, in 2014, Hsieh and Leu [24] demonstrated that Liao and Hsiao's protocol [23] is susceptible to tracing attacks. In addition, it is pointed out that the protocol is inefficient as each service server has to update its ID table periodically. Hsieh and Leu [24] then presented an improved protocol, which distributes the static registration token with the dynamic secret token. We observe that the protocols of Liao and Hsiao [23] and Hsieh and Leu [24] use only temporary secret keys (ephemeral secrets) in their authentication messages and session key computation, which can be exploited to facilitate impersonation and ESL attacks (see [18], [23], [26], [28]).

Han and Zhu [25] proposed an ECC-based MSA protocol, which was subsequently shown to be vulnerable to the ESL attack [26]. Islam [26] then proposed an ESL attack-free MSA protocol using bilinear pairings. However, Islam's protocol [26] fails to ensure the privacy of user credentials as the identity is sent in plain text. We remark that majority of the protocols in Category 2 adopt a black/white list to revoke/permit users' access privileges, without addressing the issue of revoking a server's access privileges. To be able

TABLE 1. Multi-server authentication schemes: A comparative summary.

Scheme	UC	RevS	IDTable	BWLU	TDPK	SK-S	PUCP	SACSP	PMSFRMA	IPA
Protocols in which three parties (RC, S, U) are involved in the authentication process										
Tsai [14]	Hash	No	No	Yes	Static	No	No	No	No	RC+S+U
Yoon-Yoo [15]	ECC	No	No	Yes	Static	No	No	No	No	RC+S+U
Kim *et al.* [16]	ECC	No	No	Yes	Static	No	No	No	No	RC+S+U
He-Wang [17]	ECC	No	No	Yes	Static	No	No	No	No	RC+ S+U
Odelu *et al.* [18]	ECC	Yes	Yes	List-free	Dynamic	Yes	Yes	Yes	Yes	RC+S+U
Protocols in which only two parties (S, U) are involved in the authentication process										
Chuang-Chen [19]	Hash	No	No	Yes	Static	No	No	No	No	S+U
Mishra *et al.* [20]	Hash	No	No	Yes	Static	No	Yes	No	No	S+U
Lu *et al.* [21]	ECC	No	No	Yes	Static	Yes	Yes	Yes	No	S+U
Odelu *et al.* [22]	ECC	No	No	Yes	Dynamic	Yes	Yes	Yes	No	S+U
Liao-Hsiao [23]	Pairings	No	Yes	Yes	Dynamic	No	No	No	No	S+U
Hsieh-Leu [24]	Pairings	No	Yes	Yes	Static	No	Yes	Yes	No	S+U
Han-Zhu [25]	ECC	No	Yes	Yes	Dynamic	No	No	No	No	S+U
Islam [26]	Pairings	No	Yes	Yes	Dynamic	Yes	No	Yes	No	S+U
Tseng *et al.* [27]	Pairings	Yes	Yes	List-free	Static	No	No	Yes	No	S+U
Proposed TK-SSO	ECC	Yes	Yes	List-free	Dynamic	Yes	Yes	Yes	Yes	S+U

Note: UC: Underlying cryptography; RevS: Whether server's revocation property is provided?; IDTable: Does RC maintain an identity table?; BWLU: Whether a black/white list is required for users?; TDPK: Type of distributed private keys for users and servers; SK-S: Whether SK-security is provided?; PUCP: Whether user credential privacy is ensured?; SACSP: Whether claimed security properties are sound?; PMSFRMA: Are necessary / baseline security features in multi-server environment supported?; IPA: Involved parties in the authentication.

to revoke a server's access privileges, the system manager would need to notify each user and server [27]. In 2015, Tseng *et al.* [27] constructed a list-free MSA protocol using bilinear pairings. However, their protocol does not ensure the privacy of user credential since the user's identity is (again) sent in plain text. The protocol also distributes the static authentication tokens. In other words, Tseng *et al.*'s protocol does not support the basic security features previously discussed.

In this paper, we aim to design a novel list-free MSA protocol using ECC, which offers all well known security functionalities. The functionalities of various existing multi-server authentication protocols are summarized in Table 1.

B. CONTRIBUTION AND PAPER ORGANIZATION

The key contribution in this paper is the proposed privacy preserving time-key based single sign-on authenticated key management protocol (TK-SSO) for a multi-server environment deployment. The characteristics of the proposed protocol are as follow:

- A session key is shared between a mobile user and a server in the multi-server environment, and no other parties including the registration center will learn the session key in the proposed scheme.
- User credentials' privacy is guaranteed in the proposed scheme even if the temporary information involved in the session is leaked. This feature does not exist in most of the existing multi-server schemes.
- Session key security (SK-security) is provided in the proposed scheme. The proposed scheme has the ability to withstand ephemeral secret leakage (ESL) attack and provides forward secrecy.
- Revocation property is supported in the proposed scheme, which allows one to revoke misbehaving or compromised users and servers from the system prior to their expiration dates.
- Revocation and re-registration properties using the same identity due to distribution of dynamic initial private keys are supported in the proposed scheme.
- User identity information is stored by the *RC* to prevent many logged-in users attacks as well as impersonation attacks.

The remainder of the paper is as follows. In Section II, we discuss the required preliminaries. In Section III, we present the proposed TK-SSO protocol. We prove the security of TK-SSO in Section IV, as well as using AVISPA tool in Section V. In Section VI, we evaluate the performance of TK-SSO with related protocols in the literature. Section VII concludes the paper.

II. MATHEMATICAL PRELIMINARIES

We will now briefly review relevant mathematical preliminaries used in the paper.

A. ELLIPTIC CURVE OVER PRIME FIELD GF(p)

Let p be a large prime number. A non-singular elliptic curve $\mathbb{E}_p(a, b)$ over the Galois field $GF(p)$ is defined by the equation $y^2 = x^3 + ax + b$, where $a, b \in Z_p$ and $4a^3 + 27b^2 \neq 0 \pmod p$. All points on $\mathbb{E}_p(a, b)$ and a special point $\mathcal{O}$ called the point at infinity form an additive group $\mathcal{G}$. More precisely, a well-known Hasse's theorem asserts that the number of points on $E_p(a, b)$, which is denoted by $\#E$, satisfies the following inequality:

$$p + 1 - 2\sqrt{p} \leq \#E \leq p + 1 + 2\sqrt{p}.$$

In other words, an elliptic curve $E_p(a, b)$ over Z_p has roughly p points on it.

If $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ be two points in $E_p(a, b)$, then the addition of P and Q is given by $R = (x_R, y_R) = P+Q$ and it is computed by the following rule:

$$x_R = (\mu^2 - x_P - x_Q) \pmod p,$$
$$y_R = (\mu(x_P - x_R) - y_P) \pmod p,$$

where $\mu = \begin{cases} \frac{y_Q - y_P}{x_Q - x_P} \pmod p, \text{if } P \neq -Q \\ \frac{3x_P{}^2 + a}{2y_P} \pmod p, \text{if } P = Q. \end{cases}$

Let P be a generator of $\mathcal{G}$, the ECC point multiplication or scalar multiplication kP is defined by the equation $kP = P + P + \cdots + P(k \text{ times})$.

It is well known that the below mathematical problems are computationally infeasible (hard).

Definition 1 (Elliptic Curve Discrete Logarithm Problem (ECDLP)): Given a point $Q \in \mathcal{G}$, it is hard to compute the discrete logarithm k such that $Q = kP$.

Definition 2 (Elliptic Curve Computational Diffie-Hellman Problem (ECCDHP)): Given two points $xP, yP \in \mathcal{G}$ for unknown $x, y \in Z_p$, it is hard to compute $xyP \in \mathcal{G}$.

Definition 3 (Elliptic Curve Decisional Diffie-Hellman Problem (ECDDHP)): Given three points $xP, yP, zP \in \mathcal{G}$ for unknown $x, y, z \in Z_p$, it is hard to decide whether $xyP = zP$, where $x, y, z \in Z_p^*$.

B. SYSTEM OF LINEAR EQUATIONS (SLE)

Let $c_i = a_i y + b_i z$ be a system of l linear equations (l-SLE) in y and z, where $a_i = a_j$ and $b_i = b_j$ if and only if $i = j$, for $i = 1, 2, \cdots, l$. We define the following three cases [30], [31]:

- Case 1. If both a_i and b_i are known, the equations form a system of l linear equations with two unknowns y and z. The system is solvable for y and z, and it has a unique solution.
- Case 2. If a_i (or b_i) is unknown, the equations form a system of l equations with $l + 2$ unknowns a_i (or b_i), y and z. The system is also solvable, however it has infinitely many solutions.
- Case 3. If both a_i and b_i are unknown, the equations form a system of l equations with $2l+2$ unknowns a_i, b_i, y and z. The system is still solvable, however it has infinitely many solutions.

Solving an l-SLE with $l+m$ unknowns, $m > 0$, for the correct unique solution is same as guessing randomly the m correct numbers. Thus, solving l-SLE with more than l unknowns for the correct unique solution (in short, SLEUS) is assumed to be computationally hard problem.

III. THE PROPOSED PROTOCOL

In this section, we give the details of TK-SSO for lightweight mobile devices in mobile cloud computing environment using ECC. In the proposed protocol, the *RC* (registration center) is the responsible for generating and distributing private keys of mobile users and cloud servers. The mobile user devices are assumed to be lightweight devices such as laptops and mobile phones, and these request for the services from powerful server. After mutual authentication, the cloud server and mobile user agree on a secure session key to communicate each other securely in future. Note that the *RC* does not involve in the authentication process of the proposed protocol. The notations and their meanings are listed in Table 2. We use these notations throughout this paper.

TABLE 2. Notations used in this paper.

Symbol	Description
RC	Registration center
(k, Q)	Private and public key pair of RC, where $Q = kP$
U_i and ID_i	i^{th} user and his/her identity, respectively
S_j and ID_j	j^{th} server and its identity
p	A large prime number
$\mathbb{E}_p(a, b)$	A non-singular elliptic curve $y^2 = x^3 + ax + b \pmod p$ over finite field $GF(p)$ with $4a^3 + 27b^2 \neq 0 \pmod p$
P	A base point in $\mathbb{E}_p(a, b)$ whose order is a prime q, where $q \in Z_p$ is 160-bit number
$P_1 + P_2$	Elliptic curve point addition, $P_1, P_2 \in \mathbb{E}_p(a, b)$
$x.P$	$P + P + \cdots P$ (x times), scalar multiplication, where $x \in Z_p^*$ and $P \in \mathbb{E}_p(a, b)$
G	Elliptic curve group $\{p, \mathbb{E}_p(a, b), P\}$ generated by P
$L.x$	x-coordinate of an elliptic curve point $L \in \mathbb{E}_p(a, b)$
H_u, H_s	Two secure one-way collision-resistance hash functions
Ω	Symmetric-key cryptosystem
$E_y(\cdot)/D_y(\cdot)$	Symmetric-key encryption/decryption using the key y

In TK-SSO, we adopt the similar approach for the revocation of users and servers as presented in the protocols [18], [27]. The private key consists of two parts, namely, the dynamic initial key and time update key. In TK-SSO, the initial key will be update only in the genuine case when the initial key is unexpectedly revealed, whereas most of the existing protocols distributed the static keys including Tseng *et al.*'s protocol [27]. On the other hand, the time update key is renewed periodically by the *RC* in the defined intervals, called the lifetime of the time update key. Thus, TK-SSO provides list-free authentication with revocation facility for the users as well as servers. In addition, our proposed TK-SSO provides user credentials' privacy and resists ESL attack. The proposed TK-SSO consists of the following four phases, namely, system initialization phase, registration phase, time key update phase, and authentication and key establishment phase, which are described as follows.

A. SYSTEM INITIALIZATION PHASE

In TK-SSO, the *RC* selects a non-singular elliptic curve $\mathbb{E}_p(a, b)$ and a base point P on $\mathbb{E}_p(a, b)$ which is of order 160 bits prime q in a finite field $GF(p)$, where p is a sufficiently large prime number so that ECDLP becomes intractable. Let $G = \{p, \mathbb{E}_p(a, b), P\}$ be an elliptic curve group generated by P. Further, the *RC* chooses two cryptographic collision-resistant one-way hash functions H_u, H_s: $\{0, 1\}^* \rightarrow Z_p^*$, and symmetric-key cryptosystem Ω. Both hash functions $H_u(\cdot)$ and $H_s(\cdot)$ are used in the subsequent sections and the purpose of using these hash functions are explained in those sections. Finally, the *RC* randomly chooses a random number k as its master private key, computes the corresponding public key $Q = kP$, and then publicly declares $\{G, Q, H_u, H_s, \Omega\}$.

TABLE 3. Summary of registration and time key update phases.

Registration of server	
Server S_j	RC
$\langle ID_j \rangle$ (via secure channel)	Computes $d_j = H_s(ID_j, r_j, k)$. Stores $\langle E_k(ID_j), r_j, status \in \{-1, 0, 1\}\rangle$. $\langle d_j \rangle$ (via secure channel)
Server's time key update	
	Chooses q_j. Computes $Q_j = q_j P$, $s_j = \frac{H_s(d_j, Q_j LT_j) - H_s(ID_j, LT_j, Q_j)k}{q_j}$. $\langle s_j, Q_j, LT_j \rangle$
Registration of user	
User U_i	RC
$\langle ID_i \rangle$ (via secure channel)	Computes $d_i = H_u(ID_i, r_i, k)$. Stores $\langle E_k(ID_i), r_i, status \in \{-1, 0, 1\}\rangle$. $\langle d_i \rangle$ (via secure channel)
User's time key update	
	Chooses q_i. Computes $Q_i = q_i P$, $TID_i = E_k(ID_i, LT_i)$, $s_i = \frac{H_u(d_i, Q_i, LT_i) - H_u(TID_i, Q_i, LT_i)k}{q_i}$. $\langle s_i, TID_i, Q_i, LT_i \rangle$

B. REGISTRATION PHASE

The registration process of the servers and the users for our proposed TK-SSO is discussed in the following subsections and also summarized in Table 3.

1) SERVER REGISTRATION PHASE

A server S_j selects its unique identity ID_j and sends it to the RC via a secure channel. When receiving the identity, RC randomly chooses a number r_j and computes the initial private key $d_j = H_s(ID_j, r_j, k)$. RC then stores $\langle E_k(ID_j), r_j, status \in \{-1, 0, 1\}\rangle$ in its identity table, where the status of S_j is inactive if $status = 0$, active if $status = 1$, and inactive and misbehavior/compromised if $status = -1$. Finally, RC sends the message $\langle d_j \rangle$ to S_j securely, and then S_j keeps received initial private key d_j as secret for long-term use.

2) USER REGISTRATION PHASE

A user U_i selects its unique identity ID_i and sends it to RC via a secure channel. When receiving the request, RC randomly chooses a number r_i and computes the initial private key $d_i = H_u(ID_i, r_i, k)$. RC stores $\langle E_k(ID_i), r_i, status \in \{-1, 0, 1\}\rangle$ in its identity table, where the status of U_i is inactive if $status = 0$, active if $status = 1$, and inactive and misbehavior/compromised if $status = -1$. Finally, RC sends the message $\langle d_i \rangle$ to U_i securely, and then U_i keeps secret received initial private key d_i for long-term use.

Remark 1 (Initial key update): Suppose a server S_j (or a user U_i) wants to change its initial private key for some security reasons without changing the identity ID_j (or ID_i), respectively. In this case, the server S_j (or user U_i) first needs to revoke its account, and then to send the re-registration request with the same identity to the RC. Upon receiving the re-registration request, the RC verifies the personal identity of S_j (or U_i), and then checks whether it is registered with the status as inactive. The RC then chooses a fresh random number r_j^{fresh} (or r_i^{fresh}) and issues the fresh initial private key $d_j^{fresh} = H_s(ID_j, r_j^{fresh}, k)$ (or $d_i^{fresh} = H_u(ID_i, r_i^{fresh}, k)$). Finally, the RC updates r_j (or r_i) with the corresponding r_j^{fresh} (or r_i^{fresh}) in its identity table. In this way, the re-registration can be easily performed in the proposed scheme.

C. TIME KEY UPDATE PHASE

The time update keys of the servers and users are periodically updated by RC according to the defined security requirements. The detailed process is explained in the following subsections. The time key update phase is also summarized in Table 3.

1) SERVER TIME KEY UPDATE

RC chooses a random number q_j, and computes $Q_j = q_j P$ and s_j such that $H_s(d_j, Q_j, LT_j) = s_j q_j + H_s(ID_j, LT_j, Q_j)k \pmod q$, where LT_j is the lifetime of Q_j. RC sends the time update key $\{s_j, Q_j, LT_j\}$ to S_j via a public channel. Upon receiving $\{s_j, Q_j, LT_j\}$, S_j computes $d_j' = H_s(d_j, Q_j, LT_j)$ and checks whether the condition $d_j' P = s_j Q_j + H_s(ID_j, LT_j, Q_j)Q$ holds or not. If it holds, S_j keeps the secret d_j' in its lifetime. Finally, S_j publicly declares $\{ID_j, s_j, Q_j, LT_j\}$ to provide services to valid users.

2) USER TIME KEY UPDATE

RC chooses a random number q_i, and computes the temporary identity $TID_i = E_k(ID_i, LT_i)$, $Q_i = q_i P$ and s_i such that $H_u(d_i, Q_i, LT_i) = s_i q_i + H_u(TID_i, Q_i, LT_i)k \pmod q$, where LT_i is the lifetime of $\{s_i, TID_i, Q_i\}$. RC sends the time update key $\{s_i, TID_i, Q_i, LT_i\}$ to U_i (for example, through e-mail). Upon receiving $\{s_i, TID_i, Q_i, LT_i\}$, the user U_i computes $d_i' = H_u(d_i, Q_i, LT_i)$ and checks whether the condition $d_i' P = s_i Q_i + H_u(TID_i, Q_i, LT_i)Q$ holds or not. If it holds, U_i keeps secret the time update keys $\{d_i', s_i, TID_i, Q_i, LT_i\}$ for the use of authentication in the lifetime LT_i.

Remark 2: Note that to revoke a server S_j (or user U_i), RC simply stops issuing the time update keys and sets $status = 0$ for self revoking server (or user), and sets $status = -1$ for the misbehavior/compromised revoking server (or user) in their corresponding identity table entry. As a result, the revocation property is satisfied in our proposed scheme. In the above time key update phases, the keys are generated using the ECC-based ElGammal-type digital signature [30], [31]. Thus, the hardness of time key forgery remains same as that for the ECDLP assumption.

D. AUTHENTICATION AND KEY ESTABLISHMENT PHASE

In this phase, a secure session key a user U_i and a server S_j is established after mutual authentication. It has the following steps:

TABLE 4. Summary of authentication and key establishment phase.

User U_i	Server S_j
Chooses x_u and computes $R_u = H_u(d_i', x_u)(s_jQ_j + H_s(ID_j, LT_j, Q_j)Q) = H_u(d_i', x_u)d_j'P$, $X_u = H_u(d_i', x_u)P$, and $AID_i = E_{R_u.x}(s_i, TID_i, Q_i, LT_i)$.	
$M_1 = \langle AID_i, X_u \rangle$ ⟶	
	Computes $R_s = d_j'X_u = d_j'H_u(d_i', x_u)P$, $(s_i, TID_i, Q_i, LT_i) = D_{R_s.x}(AID_i)$, Checks the validity of LT_i. Accept/reject? Chooses x_s and computes $SK_s = H_s(d_j', x_s)X_u = H_s(d_j', x_s)H_u(d_i', x_u)P$, $X_s = H_s(d_j', x_s)(s_iQ_i + H_u(TID_i, Q_i, LT_i)Q) = H_s(d_j', x_s)d_i'P$, $h_s = H_s(SK_s, X_s, X_u, TID_i, R_s)$.
	$M_2 = \langle X_s, h_s \rangle$ ⟵
Computes $SK_u = \frac{H_u(d_i', x_u)}{d_i'}X_s = H_u(d_i', x_u)H_s(d_j', x_s)P$. Checks if $h_s = H_s(SK_u, X_s, X_u, TID_i, R_u)$. Accept/reject? Computes $h_u = H_u(SK_u, X_u, X_s, Q_i, R_u)$.	
$M_3 = \langle h_u \rangle$ ⟶	
	Checks if $h_u = H_u(SK_s, X_u, X_s, Q_i, R_s)$. Accept/reject?
Both U_i and S_j agree on the shared session key $SK = SK_u = SK_s$.	

Step 1. U_i chooses a random number x_u, and computes $R_u = H_u(d_i', x_u)(s_jQ_j + H_s(ID_j, LT_j, Q_j)Q) = H_u(d_i', x_u)d_j'P$, $X_u = H_u(d_i', x_u)P$, and $AID_i = E_{R_u.x}(s_i, TID_i, Q_i, LT_i)$, where $R_u.x$ represents the x-coordinate of the ECC point R_u. Note that $R_u.x$ is 160 bits in size, and if we want to use the Advanced Encryption Standard (AES) symmetric encryption algorithm, we need to use only 128 bits from 160 bits of $R_u.x$. Finally, U_i sends the request message $M_1 = \langle AID_i, X_u \rangle$ to S_j via a public channel.

Step 2. After receiving M_1, S_j computes $R_s = d_j'X_u = d_j'H_u(d_i', x_u)P$ and retrieves TID_i, Q_i, and LT_i by decrypting AID_i using the key $R_s.x$ as $(s_i, TID_i, Q_i, LT_i) = D_{R_s.x}(AID_i)$, where $R_s.x$ represents the x-coordinate of the ECC point R_s. S_j then checks the validity of LT_i. If it is valid, S_j randomly chooses a number x_s, and computes $SK_s = H_s(d_j', x_s)X_u = H_s(d_j', x_s)\ H_u(d_i', x_u)P$, $X_s = H_s(d_j', x_s)\ (s_iQ_i + H_u(TID_i, Q_i, LT_i)Q) = H_s(d_j', x_s)d_i'P$ and $h_s = H_s(SK_s, X_s, X_u, TID_i, R_s)$. Finally, S_j sends the challenge message $M_2 = \langle X_s, h_s \rangle$ to U_i via a public channel.

Step 3. Upon receiving M_2, U_i computes the session key $SK_u = \frac{H_u(d_i', x_u)}{d_i'}X_s = H_u(d_i', x_u)\ H_s(d_j', x_s)P$, and checks whether the condition $h_s = H_s(SK_u, X_s, X_u, TID_i, R_u)$ holds or not. If it holds, U_i authenticates S_j and confirms that S_j is agreed on the shared session key is $SK = SK_u$. U_i further computes $h_u = H_u(SK_u, X_u, X_s, Q_i, R_u)$, and sends the confirmation message $M_3 = \langle h_u \rangle$ to S_j via a public channel.

Step 4. After receiving M_3, S_j checks whether the condition $h_u = H_u(SK_s, X_u, X_s, Q_i, R_s)$ holds or not. If it holds, S_j authenticates U_i and confirms that the shared session key is $SK = SK_s = SK_u$. Otherwise, S_j terminates the session.

The authentication and key establishment phase is summarized in Table 4.

IV. SECURITY ANALYSIS

In this section, as in [18] we first show that TK-SSO can provide the session-key security in the random oracle model and then prove that TK-SSO provides mutual authentication using the widely-accepted BAN logic [32]. We then show that TK-SSO is also secure against various common attacks.

The BAN logic based proof only captures the mutual authentication between two communicating entities. The random oracle model using the Real-Or-Random (ROR) model proves the semantic security of the proposed scheme for the session key (SK) security against an adversary for deriving the session key between a user U_i and a server S_j. The formal security verification using the widely-accepted AVISPA tool [18], [33] ensures that the proposed scheme is secure against the replay and man-in-the-middle attacks. Since the security analysis using ROR model and BAN logic do not capture all the attacks, we require the security analysis using AVISPA tool as well as informal (non-mathematical) security analysis.

A. ADVERSARY MODEL

Based on the Real-Or-Random (ROR) model [34], [35], we give the formal security analysis of the proposed protocol. In our proposed protocol, three parties are the user U_i, server S_j and registration center RC. A brief review of ROR model is presented as follows.

Participants. Let $\Pi_{S_j}^t$, $\Pi_{U_i}^u$ and Π_{RC}^v be the instances of t, u and v of S_j, U_i and RC, respectively, involved in our multi-server authentication environment, and these are termed as *oracles*.

Partnering. The instances $\Pi_{U_i}^u$ of U_i and $\Pi_{S_j}^t$ of S_j are the partners. We call $\Pi_{S_j}^t$ as the partner ID $pid_{U_i}^u$ of $\Pi_{U_i}^u$. The partial transcript of all the messages exchanged between U_i and S_j is unique, and it is said to be a session ID $sid_{U_i}^u$ for the current session in which $\Pi_{U_i}^u$ participates.

Freshness. $\Pi_{S_j}^t$ or $\Pi_{U_i}^u$ is said to be fresh, if the session key SK shared between U_i and S_j is not leaked to $\mathcal{A}$.

Adversary. In this model, an adversary $\mathcal{A}$ will have full control over the communication channel, and have access to the following queries:

- $Execute(\Pi^t, \Pi^u)$: Through this query, $\mathcal{A}$ can get messages exchanged between U_i and S_j. This query models a passive attack.
- $Send(\Pi^t, m)$: Through this query, $\mathcal{A}$ gets a response message after sending the message m. This query models an active attack.
- $Test(\Pi^t)$: It models the semantic security of the session key based on the indistinguishability in the ROR model [34]. A coin c is flipped at the beginning. If $c = 1$, the session key is sent to $\mathcal{A}$; otherwise ($c = 0$), a random number is sent to $\mathcal{A}$.

Semantic security of the session key. In the ROR model, $\mathcal{A}$ is challenged in an experiment to distinguish the real session key and the random number. $\mathcal{A}$ can only make one *Test* query and the response of *Test* query is consistent with

the coin c. At last, $\mathcal{A}$ outputs his/her guess c' about c. We say $\mathcal{A}$ wins the game if $c' = c$. The advantage that $\mathcal{A}$ can violate semantic security of the authenticated key agreement (AKE) protocol $\mathcal{P}$ is defined by the equation $Adv_{\mathcal{P}}^{AKE} = |2.Pr[Succ]-1|$, where *Succ* denotes the event that $\mathcal{A}$ can win the game. $\mathcal{P}$ is said to be a secure multi-server authentication protocol in ROR sense if $Adv_{\mathcal{P}}^{AKE} \leq \delta$, where $\delta > 0$ is a sufficiently small number.

Random oracle. As defined in Chang and Le [35], all the participants and the adversary $\mathcal{A}$ are provided with a collision-resistant one-way hash function $H(\cdot)$, which is modeled as a random oracle, say *Hash*. A table consists of the form (u, v) is used to simulate the oracle. Upon receiving a query $h(u)$, v is returned if a tuple (u, v) exists in the table. Otherwise, a random string v is chosen, (u, v) is stored in the table, and v is returned.

B. FORMAL SECURITY ANALYSIS USING ROR MODEL

For the formal security analysis, we utilize the difference lemma [36], [37] as follows.

Lemma 1 (Difference Lemma): Let P, Q and R denote the events defined in some probability distribution. Let $P \wedge \neg R \Leftrightarrow Q \wedge \neg R$. Then,

$$|Pr[P] - Pr[Q] \leq Pr[R].$$

Using the ROR model discussed in Section IV-A and Lemma 1, in following Theorem 1 we prove that the proposed authentication scheme is secure and provides the SK-security.

Theorem 1: Suppose $\mathcal{A}$ is an adversary running in polynomial time t against our proposed scheme $\mathcal{P}$ in the random oracle. Then, the probability that $\mathcal{A}$ breaks the SK-security of $\mathcal{P}$ is given by

$$Adv_{\mathcal{P}}^{AKE} \leq \frac{q_h^2}{|Hash|} + 2.Adv_{G_q}^{ECDDHP}(t),$$

where q_h, $|Hash|$ and $Adv_{G_q}^{ECDDHP}(t)$ denote the number of *Hash* queries made to the oracle, the output range of the hash function and the advantage of $\mathcal{A}$ in breaking the ECDDHP, respectively.

Proof: It consists of a sequence of four defined games G_i, for $i = 0, 1, 2, 3$. Suppose E_i is an event wherein the adversary $\mathcal{A}$ succeeds in guessing the bit c in game G_i and wins the game. We start with real attack game G_0 against our proposed scheme $\mathcal{P}$ and then end with game G_3 which concludes that the advantage to break SK-security of $\mathcal{P}$ is negligible.

Game G_0: This game simulates the real attack by $\mathcal{A}$ against our protocol $\mathcal{P}$. Since the bit c needs to be chosen at the beginning of G_0, by definition, we have

$$Adv_{\mathcal{P}}^{ake} = |2.Pr[E_0] - 1|. \quad (1)$$

Game G_1: This game transforms the game G_0 into G_1 with the help of simulation of $\mathcal{A}$'s eavesdropping attacks by querying $Execute(\Pi^t, \Pi^u)$ oracle. After that the *Test* oracle is queried by $\mathcal{A}$ at the end and it has to decide whether the output of the *Test* oracle is the actual session key SK $(= SK_u = SK_s)$ or a random number. The session key SK is computed by the server S_j as $SK_s = H_s(d_j', x_s)X_u = H_s(d_j', x_s)$ $H_u(d_i', x_u)P$, whereas the same session key is computed by the user U_i as $SK_u = \frac{H_u(d_i', x_u)}{d_i'}X_s = H_u(d_i', x_u)\ H_s(d_j', x_s)P$. Note that d_i' and x_u are secret to U_i, and d_j' and x_s are also secret to S_j. Therefore, without these information. eavesdropping does not increase the probability of winning the game for the adversary $\mathcal{A}$. Then, G_0 is equivalent to G_1, and thus we have,

$$Pr[E_1] = Pr[E_0]. \quad (2)$$

Game G_2: This game transforms the game G_1 into G_2 by simulating the *Send* and *Hash* oracles. G_2 models an active attack where $\mathcal{A}$ tries to decide a participant into accepting a fabricated message. Thus, $\mathcal{A}$ can make several *Hash* queries to find the collisions. Note that the messages M_1, M_2 and M_3 (in Section III-D) are associated with LT_i, Q_i, s_i, TID_i, the secrets d_i' and x_u of U_i, and the secrets d_j' and x_s of S_j. Both x_u and x_s are random numbers. Hence, there is no collision when $\mathcal{A}$ queries *Send* oracle. According to the birthday paradox [38], we have

$$|Pr[E_1] - Pr[E_2]| \leq \frac{q_h^2}{2.|Hash|}. \quad (3)$$

Game G_3: This game models an attack wherein $\mathcal{A}$ has to compute the real session key SK $(= SK_u = SK_s)$ using $X_u = H_u(d_i', x_u)P$ and $X_s = H_s(d_j', x_s)$ $(s_iQ_i + H_u(TID_i, Q_i, LT_i)Q) = H_s(d_j', x_s)d_i'P$ from the eavesdropping messages M_1 and M_2, respectively. Note that $Adv_{G_q}^{ECDDHP}(t)$ denotes the advantage of $\mathcal{A}$ wherein he/she needs to distinguish between abP, and given random $X_u = aP$ and $X_s = bP$. where $a = H_u(d_i', x_u)$ and $b = H_s(d_j', x_s)$. Therefore, we have

$$|Pr[E_2] - Pr[E_3]| \leq Adv_{G_q}^{ECDDHP}(t). \quad (4)$$

Since all the session keys are random and independent, and no information about the value of c is revealed to the adversary $\mathcal{A}$, we have

$$Pr[E_3] = 1/2. \quad (5)$$

We solve Equations (1) - (5) and use Lemma 1 to obtain the following result. Equation (1) yields

$$\frac{1}{2}.Adv_{\mathcal{P}}^{AKE} = |Pr[E_0] - \frac{1}{2}|. \quad (6)$$

Using the triangular inequality, we get,

$$\begin{aligned} |Pr[E_1] - Pr[E_3]| &\leq |Pr[E_1] - Pr[E_2]| \\ &\quad + |Pr[E_2] - Pr[E_3]| \\ &\leq \frac{q_h^2}{2.|Hash|} + Adv_{G_q}^{ECDDHP}(t). \end{aligned} \quad (7)$$

With the help of Equations (2) and (5), we have,

$$|Pr[E_0] - \frac{1}{2}| \leq \frac{q_h^2}{2.|Hash|} + Adv_{G_q}^{ECDDHP}(t). \quad (8)$$

From Equations (6) and (8), we obtain,

$$\frac{1}{2}.Adv_{\mathcal{P}}^{AKE} \leq \frac{q_h^2}{2.|Hash|} + Adv_{G_q}^{ECDDHP}(t). \quad (9)$$

Finally, Equations (6) and (9) give the final result:

$$Adv_{\mathcal{P}}^{AKE} \leq \frac{q_h^2}{|Hash|} + 2.Adv_{G_q}^{ECDDHP}(t).$$

■

C. AUTHENTICATION PROOF BASED ON BAN LOGIC

The notations used in the BAN logic are given below [32]. P and Q are the principles, X and Y are the formulas, and Z is a statement.

- $P \mid\equiv Z$: P believes Z.
- $\#(X)$: X assumed to be fresh.
- $P \mid\Rightarrow Z$: P has jurisdiction over Z.
- $P \lhd Z$: P sees the statement Z.
- $P \mid\sim Z$: P once said the statement Z.
- (X, Y) : The statement X or Y is a part of the formula (X, Y).
- $\{W\}_K$: W is encrypted using the symmetric-key K.
- $\langle X\rangle_Y$: The X is combined with shared secret Y.
- $P \overset{K}{\longleftrightarrow} Q$: K is the shared secret between P and Q, never discover to third party.
- $P \overset{X}{\rightleftharpoons} Q$: The secret X is only known to P and Q, and possibly the trusted third party by them.

Rules: We use the following four BAN logic rules in our proof:

- $R1$. Message-meaning rule: $\frac{P\mid\equiv P\overset{K}{\rightleftharpoons}Q, P\lhd\{X\}_K}{P\mid\equiv Q\mid\sim X}$ and $\frac{P\mid\equiv P\overset{Y}{\rightleftharpoons}Q, P\lhd\langle X\rangle_Y}{P\mid\equiv Q\mid\sim X}$.
- $R2$. Nonce-verification rule: $\frac{P\mid\equiv\#(X), P\mid\equiv Q\mid\sim X}{P\mid\equiv Q\mid\equiv X}$.
- $R3$. Jurisdiction rule: $\frac{P\mid\equiv Q\mid\Rightarrow X, P\mid\equiv Q\mid\equiv X}{P\mid\equiv X}$.
- $R4$. Freshness-conjuncatenation rule: $\frac{P\mid\equiv\#(X)}{P\mid\equiv\#(X,Y)}$.

Goals: We prove that the proposed scheme satisfies the following test goals that prove the proposed authentication scheme provides secure mutual authentication:

- $G_1 : U_i \mid\equiv S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$;
- $G_2 : U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$;
- $G_3 : S_j \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$;
- $G_4 : S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$.

Since the information $\{s_j, ID_j, Q_j, LT_j\}$ are the valid public key information of the server S_j within the lifetime LT_i, the key $R = R_u = R_s$ is a shared key between the user U_i and the server S_j, where $R_u = H_u(d_i', x_u)$ $(s_jQ_j + H_s(ID_j, LT_j, Q_j)Q) = H_u(d_i', x_u)d_i'P = d_j'X_u = R_s$. Thus, without any loss of generality, we assume that the key $R = R_s$ is shared key between user U_i and server S_j in that instance.

Generic form: The generic form of the communicated messages.

- From message M_1: $U_i \rightarrow S_j$: $\{s_i, TID_i, Q_i, LT_i\}_R$, $X_u = H_u(d_i', x_u)P$.
- From message M_2: $S_j \rightarrow U_i$: $X_s = H_s(d_j', x_s)d_i'P$, $\langle SK_s, X_s, X_u, TID_i\rangle_R$.
- From message M_3: $U_i \rightarrow S_j$: $\langle SK_u, X_u, X_s, Q_i\rangle_R$.

Idealized form: The idealized form of the communicated messages.

- Message $M_1 : U_i \rightarrow S_j : \{s_i, TID_i, Q_i, LT_i\}_{U_i \overset{R}{\rightleftharpoons} S_j}$.
- Message $M_2 : S_j \rightarrow U_i : \langle U_i \overset{SK}{\longleftrightarrow} S_j, X_s, X_u, TID_i\rangle_{U_i \overset{R}{\rightleftharpoons} S_j}$.
- Message $M_3 : U_i \rightarrow S_j : \langle U_i \overset{SK}{\longleftrightarrow} S_j, X_u, X_s, Q_i\rangle_{U_i \overset{R}{\rightleftharpoons} S_j}$.

Hypotheses:

- $H_1 : U_i \mid\equiv \#(X_u)$;
- $H_2 : S_j \mid\equiv \#(X_s)$;
- $H_3 : U_i \mid\equiv U_i \overset{R}{\rightleftharpoons} S_j$;
- $H_4 : S_j \mid\equiv U_i \overset{R}{\rightleftharpoons} S_j$;
- $H_5 : U_i \mid\equiv S_j \mid\Rightarrow U_i \overset{SK}{\longleftrightarrow} S_j$;
- $H_6 : S_j \mid\equiv U_i \mid\Rightarrow U_i \overset{SK}{\longleftrightarrow} S_j$.

The proof using the BAN logic rules and assumptions is as follows.

- From message M_1, we have $S_1 : S_j \lhd \{s_i, TID_i, Q_i, LT_i\}_{U_i \overset{R}{\rightleftharpoons} S_j}$.
- From H_4, S_1 and $R1$, we obtain $S_2 : S_j \mid\equiv U_i \mid\sim \langle s_i, TID_i, Q_i, LT_i\rangle$.
- From message M_2, we have $S_3 : U_i \lhd \langle U_i \overset{SK}{\longleftrightarrow} S_j, X_s, X_u, TID_i\rangle_{U_i \overset{R}{\rightleftharpoons} S_j}$.
- From H_3, S_3 and $R1$, we get $S_4 : U_i \mid\equiv S_j \mid\sim \langle U_i \overset{SK}{\longleftrightarrow} S_j, X_s, X_u, TID_i\rangle$.
- From H_1, S_4, $R2$ and $R4$, we obtain $S_5 : U_i \mid\equiv S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$. (**Goal** G_1)
- Again, from H_5, S_5 and $R3$, we get $S_6 : U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$. (*Goal* G_2)
- From message M_3, we have $S_7 : S_j \lhd \langle U_i \overset{SK}{\longleftrightarrow} S_j, X_u, X_s, Q_i\rangle_{U_i \overset{R}{\rightleftharpoons} S_j}$.
- From H_4, S_7 and $R1$, we get $S_8 : S_j \mid\equiv U_i \mid\sim \langle U_i \overset{SK}{\longleftrightarrow} S_j, X_u, X_s, Q_i\rangle$.
- From H_2, S_8, $R2$ and $R4$, we obtain $S_9 : S_j \mid\equiv U_i \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$. (*Goal* G_3)
- From H_6, S_9 and $R3$, we finally obtain $S_{10} : S_j \mid\equiv U_i \overset{SK}{\longleftrightarrow} S_j$ (*Goal* G_4)

From the goals G_1-G_4, it is proved that our proposed TK-SSO provides secure mutual authentication, and thus, the proposed protocol also prevents the man-in-the-middle attack as well as reply attack.

D. OTHER POSSIBLE ATTACKS

This section shows that our TK-SSO resists various possible well known attacks.

1) COLLABORATIVE AND KEY RECOVERY ATTACKS

In this attack, a group of users U_i^u, $u = 1, 2, \cdots, n$, and servers S_j^v, $v = 1, 2, \cdots, m$, collaborate and try to derive the system private key k using the initial private keys and time update keys. The initial private key of a user U_i^u is computed as $H_u(d_i^u, Q_i^u, LT_i^u) = s_i^u q_i^u + H_u(TID_i^u, Q_i^u, LT_i^u)k \pmod q$, where q_i^u is random secret, $TID_i^u = E_k(ID_i^u, LT_i^u)$, $Q_i^u = q_i^u P$, and s_i^u is the corresponding signature. It is clear that the time key is in the form of $c_i^u = s_i^u a_i^u + b_i^u k \pmod q$. Similarly, the server's time update key is also in the form of $c_j^v = s_j^v a_j^v + b_j^v k \pmod q$. Thus, these equations form a system of $n+m$ linear equations with $n+m+1$ unknowns k, a_i^u and a_j^v's. From Section II-B (SLE), this system is solvable and have many solutions in Z_q. This implies that computing the unique correct solution for the above SLE is same as the random guessing of the system private key k. In addition, due to the difficulty of solving ECDLP (ElGammal type signature), it is also computationally infeasible for the adversary to generate the valid time update keys $\{s_i, TID_i, Q_i, LT_i\}$ of a user and $\{s_j, Q_j, LT_j\}$ of a server without the knowledge of the system private key k. As a result, our proposed TK-SSO is secure against collaborative and key recovery attacks.

2) USER ANONYMITY

In the proposed TK-SSO, a legal user U_i's identity ID_i is included in message $AID_i = E_{R_u}(TID_i, Q_i, LT_i)$, where $R_u = H_u(x_u, d_i')(s_j Q_j + H_s(ID_j, LT_j, Q_j)Q = d_j' X_u = R_s$. In order to compute R_u $(= R_s)$, $\mathcal{A}$ needs either the pair (d_i', x_u) or the secret key d_j' of S_j. Due to the difficulty of solving ECDLP, even if $\mathcal{A}$ knows the temporary secret x_u, it is infeasible to compute the information (s_i, TID_i, Q_i, LT_i), without the knowledge of either d_i' or d_j'. Moreover, the original ID_i does not revealed to S_j, instead U_i shares a temporary identity TID_i with S_j, which is only valid in the lifetime LT_i. As a result, the proposed TK-SSO provides the user anonymity.

3) SECURE MUTUAL AUTHENTICATION

From Section IV-C, it is clear from BAN logic test goals G_1-G_4 that our proposed scheme provides secure mutual authentication between U_i and S_j. In addition, it also confirm the secure shared session key between U_i and S_j. This implies that the proposed TK-SSO also resists the replay and man-in-the-middle attacks.

4) SK-SECURITY

In our proposed TK-SSO, the session key SK is computed as $SK = \frac{H_u(d_i', x_u)}{d_i'} X_s = H_s(d_j', x_s)X_u = H_u(d_i', x_u)H_s(d_j', x_s)P$. Thus, computing SK requires any one of the pairs (d_i', x_u) and (d_j', x_s) by an adversary $\mathcal{A}$. Except these two pairs, it is a computationally hard for $\mathcal{A}$ to derive the session key SK. As a result, even if the session-specific temporary information or initial private keys are revealed, except the two pairs (d_i', x_u) and (d_j', x_s), the adversary $\mathcal{A}$ has no ability to derive the session key due to the difficulty of solving ECDLP. Therefore, the proposed TK-SSO provides the SK-security, that is, TK-SSO resists ESL attack and also provides perfect forward secrecy.

5) SERVER SPOOFING ATTACK

To impersonate a server S_j to a user U_i, an adversary $\mathcal{A}$ needs to guess the valid challenge $h_s = H_s(SK_s, X_s, X_u, TID_i, R_s)$ to be authenticated by U_i. However, computing $R_s = d_j' X_u = d_j' H_u(d_i', x_u)P$ and $SK_s = H_s(d_j', x_s)X_u = H_s(d_j', x_s)H_u(d_i', x_u)P$ without the correct pair (d_j', x_s) is computationally hard for the adversary $\mathcal{A}$ due to the difficulty of solving ECDLP. As a result, the proposed TK-SSO successfully prevents the server spoofing attack.

6) IMPERSONATION ATTACK

An adversary $\mathcal{A}$ does not have any means to get a user U_i's information in order to be authenticated by the server S_j to establish a session key. Moreover, S_j and U_i mutually authenticate each other. Thus, the attacker has no ability to derive the valid user authentication factor d_i' (or the initial key d_i) due to the difficulty of solving ECDLP. As a result, our proposed TK-SSO has the ability to prevent the impersonation attack.

V. SIMULATION FOR FORMAL SECURITY VERIFICATION USING AVISPA TOOL

We apply the widely-accepted AVISPA tool [18], [33] to simulate the proposed protocol and analyze its security functions. The AVISPA tool checks whether a security protocol is secure against replay and man-in-the-middle attacks.

AVISPA tool has been widely used in the design of security protocols because it can be used to check whether security protocols are safe or unsafe against passive and active adversaries [33]. The designed protocol models are written in the High Level Protocol Specification Language (HLPSL) [39]. AVISPA tool is used for the automated validation of Internet security-sensitive protocols and applications consisting of the four backends: (i) On-the-fly-Model-Checker (OFMC), (ii) Constraint Logic based Attack Searcher (CL-AtSe), (iii) SAT-based Model-Checker (SATMC) and (iv) Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP) The details of AVISPA tool and HLPSL specification are provided in [33] and [39].

A. HLPSL SPECIFICATION

The HLPSL implementation of the proposed scheme is done for the basic roles: *user* for user U_i, *server* for server S_j and *rc* for the *RC*. Apart from these basic roles, we have two compulsory roles: *session* for the session and *environment* for the goal and environment.

In Fig. 2, we have specified the role of the user U_i. During the user registration process, U_i first receives the start signal, and sends the registration request to the *RC* securely and then updates its state from 0 to 2, where the state is maintained by the variable *State*. After that U_i receives d_i securely from the *RC* and also updates its state from 2 to 4. During the user time key update phase, U_i receives $\{s_i, TID_i, Q_i, LT_i\}$ from the *RC* via open channel and the state is changed from 4 to 6. During

```
role user (Ui, Sj, RC : agent,
        SKuirc : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: the user Ui
played_by Ui
def=
 local State  : nat,
     IDi, IDj, K, Dj, Rj, Qj, P:  text,
     LTi, LTj, Di, Ri, Qi, TIDi: text,
     AIDi, Xu, Xs, SKs, SKu, Hs, Xxs: text,
     F : hash_func
 const user_server_xu, server_user_xs,
     s1, s2, s3, s4, s5, s6, s7 : protocol_id

init  State := 0
transition
% User registration phase
1. State = 0 ∧ RCV(start) =|>
% Send the registration request <IDi> to RC securely
State' := 2 ∧ SND({IDi}_SKuirc)
        ∧ secret({IDj}, s1, {Sj,RC})
        ∧ secret({IDi}, s4, {Ui,RC})
% Receive the message <di> from RC securely
2. State = 2 ∧ RCV({H(IDi.Ri'.K)}_SKuirc) =|>
State' := 4 ∧ secret({K}, s2, {RC})
        ∧ secret({Ri'}, s5, {RC})
% User's time key update phase
% Receive the message <TIDi, Qi, LTi> from RC vic public channel
3. State = 4 ∧ RCV({IDi.LTi}_K.F(Qi'.P).LTi) =|>
State' := 6 ∧ secret({Qi'}, s7, {RC})
% Authentication and key establishment phase
% Send the message M1 to Sj via public channel
        ∧ Xu' := new()
        ∧ AIDi' := {{IDi.LTi}_K.F(Qi'.P).
          LTi}_(F(H(H(Di.LTi).Xu').H(Dj.LTj).P))
        ∧ SND(AIDi'.F(H(H(Di.LTi).Xu').P))
% Ui has freshly generated the value xu for Sj
        ∧ witness(Ui, Sj, user_server_xu, Xu')
% Receive the message M2 from Sj via public channel
4. State = 6 ∧ RCV(F(H(H(Dj'.LTj).Xs').H(Di'.LTi).P).
             H(F(H(H(Dj'.LTj).Xs').H(H(Di'.LTi).Xu').P)
             .F(H(H(Dj'.LTj).Xs').H(Di'.LTi).P).
             F(H(H(Di'.LTi).Xu').P).
             {IDi.LTi}_K. F(H(Dj'.LTj).
             H(H(Di'.LTi).Xu').P))) =|>
% Send the message M3 to Sj via public channel
State' := 8 ∧ SND(H(F(H(H(Dj'.LTj).Xs).
              H(H(Di'.LTi).Xu').P).
              F(H(H(Di'.LTi).Xu').P).
              F(H(H(Dj'.LTj).Xs').H(Di'.LTi).P).
              F(Qi.P).
              F(H(H(Di'.LTi).Xu').H(Dj'.LTj).P) ))
% Ui's acceptance of the value xs generated for Ui by Sj
        ∧ request(Sj, Ui, server_user_xs, Xs')
end role
```

FIGURE 2. The role specification for user U_i.

```
role server (Ui, Sj, RC : agent,
        SKsjrc : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: the server Sj
played_by Sj
def=
 local State  : nat,
     IDi, IDj, K, Dj, Rj, Qj, P:  text,
     LTi, LTj, Di, Ri, Qi, TIDi: text,
     AIDi, Xu, Xs, SKs, SKu, Hs, Xxs: text,
     F : hash_func
 const user_server_xu, server_user_xs,
     s1, s2, s3, s4, s5, s6, s7 : protocol_id

init  State := 0
transition
% Server registration phase
1. State = 0 ∧ RCV(start) =|>
% Send the registration request <IDj> to RC securely
State' := 1 ∧ SND({IDj}_SKsjrc)
        ∧ secret({IDj}, s1, {Sj,RC})
% Receive <dj> from RC securely
2. State = 1 ∧ RCV({H(IDj.Rj'.K)}_SKsjrc) =|>
State' := 3 ∧ secret({K}, s2, {RC})
        ∧ secret({Rj'}, s3, {RC})
% Server's time key update phase
% Receive <Qj, LTj> from RC via public channel
3. State = 3 ∧ RCV(F(Qj'.P).LTj) =|>
State' := 5  ∧ secret({IDi}, s4, {Ui,RC})
% Authentication and key establishment phase
% Receive message M1 from Ui via public channel
4. State = 5 ∧ RCV({{IDi.LTi}_K. F(Qi'.P).LTi}_(F(H(H(Di'
          .LTi).Xu').H(Dj'.LTj).P)).
          F(H(H(Di'.LTi).Xu').P)) =|>
% Send M2 to Ui via public channel
State' := 7 ∧ Xs' := new()
        ∧ SKs' := F(H(H(Dj'.LTj).Xs').
                H(H(Di'.LTi).Xu').P)
        ∧ Xxs' := F(H(H(Dj'.LTj).Xs').H(Di'.LTi).P)
        ∧ Hs' := H(SKs'.Xxs'.F(H(H(Di'.LTi).Xu').P).
                {IDi.LTi}_K. F(H(Dj'.LTj).
                H(H(Di'.LTi).Xu').P))
        ∧ SND(Xxs'.Hs')
% Sj has freshly generated the value xs for Ui
        ∧ witness(Sj, Ui, server_user_xs, Xs')
% Receive the authentication reply message M3 from Ui
4. State = 7  ∧ RCV(H(F(H(H(Dj'.LTj).Xs).
             H(H(Di'.LTi).Xu').P).
             F(H(H(Di'.LTi).Xu').P).
             F(H(H(Dj'.LTj).Xs').H(Di'.LTi).P).
             F(Qi'.P).
             F(H(H(Di'.LTi).Xu').H(Dj'.LTj).P) )) =|>
%  Sj's acceptance of the value xu generated for Sj by Ui
State' := 9  ∧ request(Ui, Sj, user_server_xu, Xu')
end role
```

FIGURE 3. The role specification for server S_j.

the authentication & key establishment phase, U_i dispatches the message $M_1 = \langle AID_i, X_u \rangle$ to the server S_j via open channel and the declaration witness(Ui, Sj, user_server_xu, Xu') is made by U_i to indicate that U_i has freshly generated random number x_u for S_j. Finally, U_i receives the message $M_2 = \langle X_s, h_s \rangle$ from S_j via open channel. The declaration request(Sj, Ui, server_user_xs, Xs') means U_i's acceptance of the random number x_s generated for U_i by S_j. In other words, by this declaration, U_i authenticates S_j based on x_s. The declaration secret({IDj}, s1, {Sj,RC}) means that the information ID_j is only known to S_j and the RC which is characterized by the protocol id $s1$. We have also declared the secret credentials by the secret declaration. In a similar way, the roles for S_j and the RC in HLPSL specification are implemented and shown in Fig. 3 and 4, respectively.

```
role rc (Ui, Sj, RC : agent,
        SKsjrc : symmetric_key,
        SKuirc : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SND, RCV: channel(dy))
% Player: the RC
played_by RC
def=
 local State  : nat,
      IDi, IDj, K, Dj, Rj, Qj, P:  text,
      LTi, LTj, Di, Ri, Qi, TIDi: text,
      AIDi, Xu, Xs, SKs, SKu, Hs, Xxs: text,
      F : hash_func
 const user_server_xu, server_user_xs,
      s1, s2, s3, s4, s5, s6, s7 : protocol_id

init  State := 0
transition
% Server registration phase
% Receive the registration request <IDj> from Sj securely
1. State  = 0 ∧ RCV({IDj}_SKsjrc) =|>
State' := 1 ∧ secret({IDj}, s1, {Sj,RC})
% Send the message <dj> to Sj securely
          ∧ Rj' := new()
          ∧ Dj' := H(IDj.Rj'.K)
% Send the message <dj> to Sj securely
          ∧ SND({Dj'}_SKsjrc)
          ∧ secret({K}, s2, {RC})
          ∧ secret({Rj'}, s3, {RC})
% User registration phase
% Receive the registration request <IDi> from Ui securely
2. State = 1 ∧ RCV({IDi}_SKuirc)  =|>
State' := 2 ∧ Ri' := new()
          ∧ Di' := H(IDi.Ri'.K)
% Send the message <di> to Ui securely
          ∧ SND({Di'}_SKuirc)
          ∧ secret({IDi}, s4, {Ui,RC})
          ∧ secret({Ri'}, s5, {RC})
% Server's time key update phase
% Send the message <Qj, LTj> to Sj via public channel
          ∧ Qj' := new()
          ∧ SND(F(Qj'.P).LTj)
          ∧ secret({Qj'}, s6, {RC})
% User's time key update phase
% Send the message <TIDi, Qi, LTi> to Ui via public channel
          ∧ Qi' := new()
          ∧ TIDi' := {IDi.LTi}_K
          ∧ SND(TIDi'.F(Qi'.P).LTi)
          ∧ secret({Qi'}, s7, {RC})
end role
```

FIGURE 4. The role specification for *RC*.

Finally, we have provided the specifications in HLPSL for the roles of session, and goal and environment in Fig. 6 and 7, respectively. In the session goal, all the basic roles: *user*, *server* and rc are instanced with concrete arguments. The top-level role, known as *environment*, defines in the specification of HLPSL that contains the global constants and a composition of one or more sessions. It is worth noticing that the intruder (always indicated by the variable *i*) also plays some roles as legitimate users. Hence, *i* also takes participation in the execution of the protocol as a concrete session.

B. ANALYSIS OF SIMULATION RESULTS

We have simulated our proposed TK-SSO using the widely-accepted OFMC and CL-AtSe backends [40] for the formal security verification under the SPAN (Security Protocol ANimator for AVISPA) [41]. The simulation results are reported in Fig. 5. For the formal security verification, we have chosen the widely-accepted back-ends: OFMC and CL-AtSe for the execution tests and a bounded number of sessions model checking. The following verifications are performed in our implementation:

```
role session (Ui, Sj, RC : agent,
        SKsjrc : symmetric_key,
        SKuirc : symmetric_key,
        % H is one−way hash function
        H: hash_func)
def=
 local  S1, S2, S3, R1, R2, R3 : channel (dy)
 composition
    server (Ui, Sj, RC, SKsjrc, H, S1, R1)
 ∧ user (Ui, Sj, RC, SKuirc, H, S2, R2)
 ∧ rc (Ui, Sj, RC, SKsjrc, SKuirc, H, S3, R3)

end role
```

FIGURE 5. The results of the analysis using OFMC and CL-AtSe backends

- *Replay attack check:* In HLPSL, the OFMC and CL-AtSe back-ends check whether the legitimate agents can execute the specified protocol by performing a search of a passive intruder. The backends provide the intruder the knowledge of some normal sessions between the legitimate agents. The test results shown in Fig. 5 clearly shows that our scheme is secure against the replay attack.
- *Dolev-Yao model check:* For the Dolev-Yao model check, OFMC and CL-AtSe backends verifies if there is any man-in-the-middle attack possible by the intruder in the system.

Under the OFMC backend, the depth for the search is 12, where 3344 nodes have been searched in 15.21 seconds and output of the results are shown in Fig. 5. Under the CL-AtSe backend, 5 states were analyzed, where one state was reachable. The translation took 0.23 seconds and the computation required 0.01 seconds. It is evident from the reported results that our scheme fulfills the design properties, and our scheme is secure under the test of AVISPA using OFMC and CL-AtSe backends with the bounded number of sessions.

From the results, it is evident that our scheme is secure against replay and man-in-the-middle attacks.

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our scheme with Han and Zhu's scheme [25], Islam's scheme [26] and Tseng *et al.*'s scheme [27].

The following notations are used in the evaluation:

- T_e: time to execute a bilinear map.
- T_m: time to execute a point multiplication in a bilinear pairing group.
- T_h: time to execute a map-to-point hash function in a bilinear pairing group.
- T_{ecm}: time to execute a point multiplication in an elliptic curve group.

```
role environment()
def=
  const ui, sj, rc: agent,
        sksjrc, skuirc : symmetric_key,
        h, f : hash_func,
        k, idi, idj, ri, rj, qi, qj, lti, ltj: text,
        user_server_xu, server_user_xs,
        s1, s2, s3, s4, s5 : protocol_id
  intruder_knowledge = {ui, sj, rc, h, f, lti, ltj}
  composition
   session(ui, sj, rc, sksjrc, skuirc, h)
/\ session(i, sj, rc, sksjrc, skuirc, h)
        /\ session(ui, i, rc, sksjrc, skuirc, h)
        /\ session(ui, sj, i, sksjrc, skuirc, h)
end role

goal
  secrecy_of s1
  secrecy_of s2
  secrecy_of s3
  secrecy_of s4
  secrecy_of s5
  secrecy_of s6
  secrecy_of s7
  authentication_on user_server_xu
  authentication_on server_user_xs
end goal
environment()
```

FIGURE 6. The role specification for session.

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  C:\progra~1\SPAN\testsuite
    \results\auth.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 15.21s
  visitedNodes: 3344 nodes
  depth: 12 plies
```

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL
PROTOCOL
  C:\progra~1\SPAN\testsuite
    \results\auth.if
GOAL
  As Specified
BACKEND
  CL-AtSe

STATISTICS
  Analysed  : 5 states
  Reachable  : 1 states
  Translation: 0.23 seconds
  Computation: 0.01 seconds
```

FIGURE 7. The role specification for goal and environment.

- T_{add}: time to execute a point addition in an elliptic curve group or bilinear pairing group.
- T_H: time to execute a one-way hash function.
- T_Ω: time to execute a symmetric key encryption/decryption operation.

Similar to the approaches in [25], [27], and [42], we use the following security parameters to achieve the 1024-bit level security. For pairing based protocols, we use a supper singular or non-supper singular curve over a finite field $GF(p)$, with p a prime of 512 bits, and a large prime order of $q = 160$ bits. On the other hand, for ECC-based protocols, we use the ECC group on the Koblitz elliptic curve $y^2 = x^3 + ax + b$ defined over $GF(2^{163})$, where $a = 1$ and b is an 163-bit random prime. In addition, we assume the approximate execution timings for the various related cryptographic operations for a user and a server implemented separately on the Philips HiPersmart card and the Pentium IV computer with the maximum clock speeds of 36MHz and 3GHz, respectively (see Table 5, also reported in [42]). We use the execution timings according to the various cryptographic operations listed in Table 5 in order to evaluate the performance of the protocols.

TABLE 5. Execution timings of various cryptographic operations.

Entity	T_e	T_m	T_h	T_{ecm}	T_{add}	T_H
Server	$3.16ms$	$1.17ms$	$< 1ms$	$0.55ms$	$0.1ms$	$0.01ms$
User	$0.38s$	$0.13s$	$< 0.1s$	$0.06s$	$0.01s$	$0.001s$

TABLE 6. Computation cost comparison.

Scheme	Computation cost for user	Computation cost for server
Han-Zhu	$5T_{ecm} + 2T_{add} + 4T_H$ $\approx 0.324s$	$5T_{ecm} + 2T_{add} + 4T_H$ $\approx 2.99ms$
Islam	$2T_e + 3T_m + 2T_h$ $+2T_{add} + T_H$ $\approx 1.371s$	$2T_e + 4T_m + 3T_h$ $+2T_{add} + T_H$ $\approx 14.21ms$
Tseng *et al.*	$T_e + 3T_m + 2T_h$ $+2T_{add} + 3T_H$ $\approx 0.993s$	$T_e + 2T_m + 2T_h$ $+T_{add} + 3T_H$ $\approx 7.63ms$
Proposed TK-SSO	$4T_{ecm} + T_{add}$ $+4T_H + T_\Omega$ $\approx 0.251s$	$4T_{ecm} + T_{add}$ $+4T_H + T_\Omega$ $\approx 2.35ms$

TABLE 7. Communication cost comparison.

Scheme	Total bandwidth (Communication cost in bits)
Han-Zhu	1758 (two messages)
Islam	4288 (two messages)
Tseng *et al.*	2384 (three messages)
Proposed TK-SSO	1414 (three messages)

The computational cost and the approximate protocol execution timings during the authentication and key establishment phase are listed in Table 6. For simplicity, we assume that the execution time of the symmetric-key encryption/decryption is approximately equal to the execution time required for one-way hash function, i.e. $T_\Omega \approx T_H$. In the proposed TK-SSO, the computational cost required for a user U_i is $4T_{ecm} + T_{add} + 4T_H + T_\Omega$ operations and a server S_j is $4T_{ecm} + T_{add} + 4T_H + T_\Omega$ operations. Furthermore, in our proposed TK-SSO, the approximate protocol execution timings for a user U_i and a server S_j are $0.251s$ and $2.35ms$, respectively. It is observed that the proposed TK-SSO significantly reduces the computation costs in both authentication and key establishment phases.

In Table 7, the bandwidth requirements in the protocols in both authentication and key establishment phases are evaluated. In order to estimate the approximate communication cost (total bandwidth) to establish a secure session, we assume the following bit lengths (following to the approach in [25]). The bit lengths of an identity, life-time, symmetric-key plaintext/ciphertext block, and one-way hash function are 16 bits, 32 bits, 128 bits (for example, if we use AES-128) and 160 bits, respectively. The bit length of

a point on a pairing curve is $2 \times 512 = 1024$ bits, whereas an ECC point requires $2 \times 163 = 326$ bits. The communication bandwidth required in the proposed TK-SSO is computed as follows: the message $M_1 = \langle AID_i, X_u \rangle$ needs approximately $\lceil (160 + 128 + 326 + 32)/128 \rceil \times 128 + 326 = 768$ bits and the message $M_2 = \langle X_s, h_s \rangle$ requires $326 + 160 = 486$ bits, whereas the message $M_3 = \langle h_u \rangle$ needs 160 bits. From Table 6, it is observed that the total bandwidth required in the proposed TK-SSO is approximately 1414 bits, whereas the bandwidth requirements in Han-Zhu's scheme, Islam's scheme and Tseng *et al.*'s scheme are 1758 bits, 4288 bits and 2384 bits, respectively.

Finally, Table 1 outlines the supported security features in the protocols, and it can be observed that the protocols of Islam and Tseng *et al.* are based on bilinear pairings, whereas TK-SSO and the protocol of Han and Zhu are ECC-based. Han-Zhu's scheme and Islam's scheme do not support the revocability for the servers. In Tseng *et al.*'s scheme, the distributed private keys for users and servers are static, whereas in Han-Zhu's scheme, Islam's scheme and TK-SSO the distributed private keys for users and servers are dynamic. Moreover, Han-Zhu's scheme and Tseng *et al.*'s scheme do not provide SK-security or ensure user credentials privacy. It is also noted that Han-Zhu's scheme, Islam's scheme and Tseng *et al.*'s scheme do not support all necessary security features required in a multi-server environment, unlike TK-SSO.

VII. CONCLUSION

Single Sign-on authentication is an active research topic in distributed computing, partly due to the ongoing developments in wireless networks and mobile cloud computing. In such environment, users need to access the remote servers for various services using lightweight devices (e.g. Android and iOS devices).

In this paper, we proposed a novel ECC-based revocable privacy preserving protocol for lightweight mobile devices in the mobile cloud computing environment, TK-SSO. We also pointed out that majority of the existing schemes do not provide the necessary security features required in a multi-server deployment, while TK-SSO is designed to support these features. For example, we demonstrated that TK-SSO supports revocation which allows one to revoke misbehaving or compromised users or servers from the system prior to their expiration dates. We also proved the security of TK-SSO mathematically and using BAN logic, and evaluated the performance of the proposal against existing related protocols. The evaluations showed that both computation and communication costs in TK-SSO are significantly less that those in the related schemes.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers and the Associate Editor for their valuable feedback on the paper which helped us to improve its quality and presentation.

REFERENCES

[1] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 337–368, 1st Quart., 2014.

[2] D. Wang, D. He, P. Wang, and C.-H. Chu, "Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment," *IEEE Trans. Depend. Sec. Comput.*, vol. 12, no. 4, pp. 428–442, Jul./Aug. 2015.

[3] K.-K. R. Choo, "High tech criminal threats to the national information infrastructure," *Inf. Secur. Tech. Rep.*, vol. 15, no. 3, pp. 104–111, 2010.

[4] K.-K. R. Choo, "The cyber threat landscape: Challenges and future research directions," *Comput. Secur.*, vol. 30, no. 8, pp. 719–731, 2011.

[5] N. H. A. Rahman and K.-K. R. Choo, "A survey of information security incident handling in the cloud," *Comput. Secur.*, vol. 49, pp. 45–69, Mar. 2015.

[6] K.-K. R. Choo, *Secure Key Establishment*, vol. 41. Boston, MA, USA: Springer, 2008.

[7] K.-K. R. Choo, "Organised crime groups in cyberspace: A typology," *Trends Org. Crime*, vol. 11, no. 3, pp. 270–295, 2008.

[8] M. Ge, K.-K. R. Choo, H. Wu, and Y. Yu, "Survey on key revocation mechanisms in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 63, pp. 24–38, Mar. 2016.

[9] J.-L. Tsai and N.-W. Lo, "A privacy-aware authentication scheme for distributed mobile cloud computing services," *IEEE Syst. J.*, vol. 9, no. 3, pp. 805–815, Sep. 2015.

[10] A. Armando, R. Carbone, L. Compagna, J. Cuéllar, G. Pellegrino, and A. Sorniotti, "An authentication flaw in browser-based single sign-on protocols: Impact and remediations," *Comput. Secur.*, vol. 33, pp. 41–58, Mar. 2013.

[11] Google. (2008). *SAML Single Sign-On (SSO) Service for Google Apps*. Accessed: Aug. 2017. [Online]. Available: https://developers.google.com/google-apps/sso/saml_reference_implementation?hl=zh-tw

[12] OpenID Foundation. *The OpenID User Interface Extension Best Practices for Identity Providers 2009*. Accessed: Aug. 2017. [Online]. Available: http://wiki.openid.net/w/page/12995153/Details-of-UX-Best-Practices-for-OPs

[13] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, "Recommendation for key management part 1: General (revision 3)," *NIST Special Publication*, vol. 800, no. 57, pp. 1–147, 2012.

[14] J.-L. Tsai, "Efficient multi-server authentication scheme based on one-way hash function without verification table," *Comput. Secur.*, vol. 27, nos. 3–4, pp. 115–121, 2008.

[15] E.-J. Yoon and K.-Y. Yoo, "Robust biometrics-based multi-server authentication with key agreement scheme for smart cards on elliptic curve cryptosystem," *J. Supercomput.*, vol. 63, no. 1, pp. 235–255, Jan. 2013.

[16] H. Kim, W. Jeon, K. Lee, Y. Lee, and D. Won, "Cryptanalysis and improvement of a biometrics-based multi-server authentication with key agreement scheme," in *Proc. 12th Int. Conf. Comput. Sci. Appl. (ICCSA)*, Salvador de Bahia, Brazil, 2012, pp. 391–406.

[17] D. He and D. Wang, "Robust biometrics-based authentication scheme for multiserver environment," *IEEE Syst. J.*, vol. 9, no. 3, pp. 816–823, Sep. 2015.

[18] V. Odelu, A. K. Das, and A. Goswami, "A secure biometrics-based multi-server authentication protocol using smart cards," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 1953–1966, Sep. 2015.

[19] M.-C. Chuang and M. C. Chen, "An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics," *Expert Syst. Appl.*, vol. 41, no. 4, pp. 1411–1418, Mar. 2014.

[20] D. Mishra, A. K. Das, and S. Mukhopadhyay, "A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards," *Expert Syst. Appl.*, vol. 41, no. 18, pp. 8129–8143, 2014.

[21] Y. Lu, L. Li, H. Peng, and Y. Yang, "A biometrics and smart cards-based authentication scheme for multi-server environments," *Secur. Commun. Netw.*, vol. 8, no. 17, pp. 3219–3228, 2015.

[22] V. Odelu, A. K. Das, and A. Goswami, "A secure and efficient ECC-based user anonymity preserving single sign-on scheme for distributed computer networks," *Secur. Commun. Netw.*, vol. 8, no. 9, pp. 1732–1751, 2015.

[23] Y.-P. Liao and C.-M. Hsiao, "A novel multi-server remote user authentication scheme using self-certified public keys for mobile clients," *Future Generat. Comput. Syst.*, vol. 29, no. 3, pp. 886–900, 2013.

[24] W.-B. Hsieh and J.-S. Leu, "An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures," *J. Supercomput.*, vol. 70, no. 1, pp. 133–148, 2014.

[25] W. Han and Z. Zhu, "An ID-based mutual authentication with key agreement protocol for multiserver environment on elliptic curve cryptosystem," *Int. J. Commun. Syst.*, vol. 27, no. 8, pp. 1173–1185, 2014.
[26] S. H. Islam, "A provably secure ID-based mutual authentication and key agreement scheme for mobile multi-server environment without ESL attack," *Wireless Pers. Commun.*, vol. 79, no. 3, pp. 1975–1991, 2014.
[27] Y. M. Tseng, S. S. Huang, T. T. Tsai, and J. H. Ke, "List-free ID-based mutual authentication and key agreement protocol for multiserver architectures," *IEEE Trans. Emerg. Topics Comput.*, vol. 4, no. 1, pp. 102–112, Jan. 2016.
[28] R. C. Wang, W. S. Juang, and C. L. Lei, "User authentication scheme with privacy-preservation for multi-server environment," *IEEE Commun. Lett.*, vol. 13, no. 2, pp. 157–159, Feb. 2009.
[29] D. He, J. Bu, S. Chan, C. Chen, and M. Yin, "Privacy-preserving universal authentication protocol for wireless communications," *IEEE Trans. Wireless Commun.*, vol. 10, no. 2, pp. 431–436, Feb. 2011.
[30] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.
[31] L. Harn and Y. Xu, "Design of generalised ElGamal type digital signature schemes based on discrete logarithm," *Electron. Lett.*, vol. 30, no. 24, pp. 2025–2026, 1994.
[32] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," *Proc. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 426, no. 1871, pp. 233–271, 1989.
[33] AVISPA. *Automated Validation of Internet Security Protocols and Applications*. Accessed: Jun. 2017. [Online]. Available: http://www.avispa-project.org/
[34] M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proc. 8th Int. Conf. Theory Pract. Public Key Cryptogr. (PKC)*, Les Diablerets, Switzerland, 2005, pp. 65–84.
[35] C.-C. Chang and H.-D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 357–366, Jun. 2016.
[36] V. Shoup, "Sequences of games: A tool for taming complexity in security proofs," in *Proc. IACR Cryptol. ePrint Archive*, 2004, p. 332.
[37] T.-F. Lee, "Provably secure anonymous single-sign-on authentication mechanisms using extended Chebyshev chaotic maps for distributed computer networks," *IEEE Syst. J.*, to be published, doi: 10.1109/JSYST.2015.2471095.
[38] V. Boyko, P. MacKenzie, and S. Patel, "Provably secure password-authenticated key exchange using Diffie-Hellman," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn. (EUROCRYPT)*, 2000, pp. 156–171.
[39] D. von Oheimb, "The high-level protocol specification language hlpsl developed in the eu project avispa," in *Proc. 3rd APPSEM II (Appl. Semantics II) Workshop (APPSEM)*, Frauenchiemsee, Germany, 2005, pp. 1–17.
[40] D. Basin, S. Mödersheim, and L. Vigano, "OFMC: A symbolic model checker for security protocols," *Int. J. Inf. Secur.*, vol. 4, no. 3, pp. 181–208, 2005.
[41] AVISPA. *SPAN: Security Protocol Animator for AVISPA*. Accessed: Aug. 2017. [Online]. Available: http://www.avispa-project.org/
[42] M. Scott, N. Costigan, and W. Abdulwahab, "Implementing Cryptographic Pairings on Smartcards," in *Proc. 8th Int. Workshop Cryptograph. Hardw. Embedded Syst. (CHES)*, Yokohama, Japan, 2006, pp. 134–147.

VANGA ODELU received the M.Tech. and Ph.D. degrees in computer science and data processing from IIT Kharagpur, India. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Indian Institute of Information Technology Chittoor, Sricity, India. He has authored over 44 papers in international journals and conferences in his area of research. His research interests include cryptography, network security, hierarchical access control, remote user authentication, cloud computing security, and smart grid security.

ASHOK KUMAR DAS (M'17) received the M.Sc. degree in mathematics, the M.Tech. degree in computer science and data processing, and the Ph.D. degree in computer science and engineering from IIT Kharagpur, India. He is currently an Assistant Professor with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, India. His current research interests include cryptography, wireless sensor network security, hierarchical access control, security in vehicular ad hoc networks, smart grid, Internet of Things (IoT), cyber-physical systems and cloud computing, and remote user authentication. He has authored over 150 papers in international journals and conferences in the above areas. Some of his research findings are published in top cited journals, such as the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON SMART GRID, the IEEE INTERNET OF THINGS JOURNAL, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON CONSUMER ELECTRONICS, the IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS, the *IEEE Consumer Electronics Magazine*, the IEEE ACCESS, the *IEEE Communications Magazine*, *Future Generation Computer Systems*, and the *Journal of Network and Computer Applications*. He has served as a Program Committee Member for many international conferences. He was a recipient of the Institute Silver Medal from IIT Kharagpur. He is on the Editorial Board of the *KSII Transactions on Internet and Information Systems* and the *International Journal of Internet Technology and Secured Transactions* (Inderscience), and a Guest Editor of the *Computers and Electrical Engineering* (Elsevier) for the special issue on big data and IoT in e-healthcare.

KIM-KWANG RAYMOND CHOO (SM'15) received the Ph.D. degree in information security from the Queensland University of Technology, Australia, in 2006. He currently holds the Cloud Technology Endowed Professorship with the Department of Information Systems and Cyber Security, The University of Texas at San Antonio. He is also an Adjunct Associate Professor of cyber security and forensics with the University of South Australia, Australia. He is a fellow of the Australian Computer Society. He serves on the Editorial Board of *Computers and Electrical Engineering*, *Cluster Computing*, *Digital Investigation*, the IEEE ACCESS, the IEEE CLOUD COMPUTING, the *IEEE Communications Magazine*, *Future Generation Computer Systems*, the *Journal of Network and Computer Applications*, *PLoS ONE*, and *Soft Computing*. He also serves as the Special Issue Guest Editor of the *ACM Transactions on Embedded Computing Systems* (2017; DOI: 10.1145/3015662), the *ACM Transactions on Internet Technology* (2016; DOI: 10.1145/3013520), *Computers and Electrical Engineering* (2017; DOI: 10.1016/j.compeleceng.2017.09.023), *Digital Investigation* (2016; DOI: 10.1016/j.diin.2016.08.003), *Future Generation Computer Systems* (2016; DOI: 10.1016/j.future.2016.04.017, 2018; DOI: 10.1016/j.future.2017.09.014), the IEEE CLOUD COMPUTING (2015; DOI: 10.1109/MCC.2015.84), the IEEE NETWORK (2016; DOI: 10.1109/MNET.2016.7764272), the IEEE TRANSACTIONS ON CLOUD COMPUTING (2017; DOI: 10.1109/TCC.2016.2582278), the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING (2017; DOI: 10.1109/TDSC.2017.2664183), the *Journal of Computer and System Sciences* (2017; DOI: 10.1016/j.jcss.2016.09.001), *Multimedia Tools and Applications* (2017; DOI: 10.1007/s11042-016-4081-z), *Personal and Ubiquitous Computing* (2017; DOI: 10.1007/s00779-017-1043-z), *Pervasive and Mobile Computing* (2016; DOI: 10.1016/j.pmcj.2016.10.003), and *Wireless Personal Communications* (2017; DOI: 10.1007/s11277-017-4278-0).

NEERAJ KUMAR (M'16–SM'17) received the Ph.D. degree in computer science and engineering from Shri Mata Vaishno Devi University, Katra, India, in 2009. He was a Post-Doctoral Research Fellow with Coventry University, Coventry, U.K. He is currently an Associate Professor with the Department of Computer Science and Engineering, Thapar University, Patiala, India. He has authored over 160 technical research papers published in leading journals and conferences from the IEEE, Elsevier, Springer, and John Wiley. Some of his research findings are published in top cited journals, such as the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the IEEE TRANSACTIONS ON CONSUMER ELECTRONICS, the IEEE NETWORK, the IEEE COMMUNICATIONS, the IEEE WIRELESS COMMUNICATIONS, the IEEE INTERNET OF THINGS JOURNAL, and the IEEE SYSTEMS JOURNAL. He has guided many research scholars leading to Ph.D. and M.E./M.Tech. degrees. He is on the Editorial Board of the *Journal of Network and Computer Applications* (Elsevier) and the *International Journal of Communication Systems* (Wiley).

YOUNGHO PARK (M'17) received the B.S., M.S., and Ph.D. degrees in electronic engineering from Kyungpook National University, Daegu, South Korea, in 1989, 1991, and 1995, respectively. From 1996 to 2008, he was a Professor with the School of Electronics and Electrical Engineering, Sangju National University, South Korea. From 2003 to 2004, he was a Visiting Scholar with the School of Electrical Engineering and Computer Science, Oregon State University, USA. He is currently a Professor with the School of Electronics Engineering, Kyungpook National University. His research interests include computer networks, multimedia, and information security.