IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 10, 2019, accepted August 28, 2019, date of publication September 2, 2019, date of current version September 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2938898

A Certificateless Verifiable Strong Designated Verifier Signature Scheme

SHU HAN[1], MANDE XIE[1], BAILIN YANG[1], RONGXING LU[2], HAIYONG BAO[1], JIANHONG LIN[3], HAI-BO HONG[1], MIAN-XUE GU[1], AND SONG HAN[1]

[1]School of Computer and Information Engineering, Zhejiang Gognshang University, Hangzhou 310018, China
[2]Faculty of Computer Science, University of New Brunswick, Saint John, NB E2L 4L5, Canada
[3]Zhejiang Ponshine Information Technology Company Ltd., Hangzhou 310012, China

Corresponding authors: Bailin Yang (ybll@mail.zjsu.edu.cn) and Song Han (hansongau@gmail.com)

This work was supported in part by the National Key Research and Development Program Project of China under Grant 2017YFB1401304, in part by the Key Research and Development Program Project of Zhejiang Province under Grant 2019c01004, and in part by the Key Research and Development Program Project of Hangzhou under Grant 20182011A46.

ABSTRACT Certificateless strong designated verifier signature schemes have realized the merit of CL-PKC against the traditional strong designated verifier signatures. However, when the signer and the designated verifier disagree with the signature, existing schemes cannot distinguish the original signature from the signature transcript. In addition, errors or malicious actions introduced by the designated verifier may lead to the failure of signature verification. To solve these issues, we propose a certificateless verifiable strong designated verifier signature scheme. When disputes arise between the signer and the verifier, the scheme can effectively prevent the signer from denying the signature generated by the signer, as well as the designated verifier from denying valid signatures or invalid ones. Our scheme does not rely on bilinear pairings. The proposed scheme satisfies the requirements of verifiability, unforgeability, non-delegability, non-transferability and signer ambiguity. We also provide formal security proof in the random oracle model for the proposed scheme.

INDEX TERMS Certificateless signature, strong designated verifier signature, ECDLP, verifiable.

I. INTRODUCTION

With the rapid development of mobile technology, Internet of Things and wireless sensor network, it is particularly important to ensure communication security in these applications. Secure cryptographic protocols could address communication security problems above. Shamir [1] introduced ID-based cryptosystem (IBC) in 1984. In IBC, users can use unique identifiers as public keys. However, this requires full trust in the public key generation (PKG). In 2003, Al-Riyami and Paterson [2] proposed certificateless public key cryptography (CL-PKC). The semi-trusted third-party key generation center (KGC) replaces the trusted third party PKG in the identity-based cryptosystem. KGC only generates partial private keys for users, while users independently generate their own public key and private key by using the partial private keys provided by KGC and secret values selected by herself. In this way, KGC still maintains its own master public key and master private key, manages part of the user's private key and cannot obtain the user's secret key, thus solving the key escrow problem [3].

In the designated verifier signature scheme, only the designated verifier can verify the signature, which is different from the standard signature algorithm where anyone could verify the signature. Since the signature is shared by the signer and the designated verifier, when an adversary intercepts the signature before the signature is received, the adversary could judge who is the signer and who is the verifier. To solve this problem, the concept of strongly designated verifier is proposed. The designated verifier in the signature scheme needs to generate a transcript which cannot be distinguished from the original signature. Only the verifier can verify the validity of the original signature. When the signer and the designated verifier disagree with the signature, most strong designated verifier signature (SDVS) schemes cannot distinguish the original signature from the transcript. In other words, these schemes are deniable. In addition, in the practical applications, there may be some errors or malicious actions in the

The associate editor coordinating the review of this article and approving it for publication was Yi Qian.

verification process introduced by the designated verifier, which may lead to the failure of signature verification, e.g. a malicious verifier who deliberately denies the validity of a valid signature or the verifier makes mistakes such as entering a wrong signature in the signature verification. These errors or malicious actions could lead to invalidity of valid signatures. It is unfair to the signer.

Consider such a scenario: User A may purchase a software from Company B and enter the corresponding serial number to the system to activate the software. However, the signature verification failed and the software cannot be activated. Software Company B convinced that its signature is valid. User A also insists on having purchased a genuine software, and carried out the signature verification correctly. In other words, the signer and the verifier dispute the signature. At this moment, if there is no credible arbiter to judge the validity of the signature, it is unfair for both of them. Therefore, we propose a verifiable certificateless strong designated verifier signature scheme to solve the problem effectively.

To address the above issues, a certificateless verifiable strong designated verifier signature scheme without bilinear pairings is proposed in this paper. Our contributions in this paper are as follows.

1) We propose a certificateless verifiable strong designated verifier signature scheme without bilinear pairings.
2) In case of disputes between the signer and the verifier, the scheme can effectively prevent the signer from denying the signature generated by herself/himself, as well as the designated verifier from denying valid signatures or invalid ones.
3) The proposed scheme satisfies the requirements of verifiability, unforgeability, non-delegability, non-transferability and signer ambiguity. And we prove it in the random oracle model under the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP).

The rest of the paper is organized as follows: Section II reviews related work. Section III provides preliminaries; Section IV presents security models for the proposed scheme; Section V provides the proposed scheme. Section VI gives a formal security proof; Section VII presents performance comparison. Section VIII concludes the paper.

II. RELATED WORK

A. CERTIFICATELESS SIGNATURE

In traditional public key cryptography (PKC), it is necessary to verify the user's public key before using it, because it is necessary to ensure that malicious third parties cannot tamper with or replace the user's public key.

To solve this problem, the general method is to bind users and their public keys by issuing certificates by certification authority (CA). However, this greatly increases the use and management cost of certificate management (certificate revocation, storage, distribution and verification, etc.).

In 1984, Shamir [1] first introduced ID-based cryptosystem (IBC) to solve the heavy certificate management in public key infrastructure (PKI). However, IBC has an unavoidable key escrow problem, that is, the private key generator (PKG) has the private key of all users, and the PKG must be completely trusted. In 2003, Al-Riyami and Paterson [2] first proposed certificateless public key cryptography (CL-PKC). Following the pioneering work of Al-Riyami and Paterson, Huang *et al.* [4] reported a security loophole in their scheme. In 2004, Yum and Lee [5] proposed a general structure of certificateless signature, but Hu *et al.* [6], [7] found that this structure was insecure for type I adversary. The type I adversary is a malicious user who cannot access the master key and the target user's partial private key, but can replace the public key of any user. They provided an improved scheme and analyzed the security in a simplified security model. In 2009, Shim [8] pointed out that certificateless signature schemes are generally forgeable against type I adversaries, who can obtain valid signatures by replacing the users' public key. This means that security of signature schemes in one system setting cannot guarantee the security of signature schemes in another system setting.

Later, Huang et al. defined new and different formal security models [9], [10]. The adversaries can be divided into normal, strong and super levels according to their attack power. Subsequently, Tso *et al.* [11] also proposed another security model. Choi *et al.* [12] proposed a new short certificateless signature scheme which can resist super-level adversaries. It is the first certificateless signature scheme to satisfy the strongest security level and the shortest signature length.

Some different types of certificateless signature schemes have been proposed, including threshold signature, proxy signature, aggregate signature, designated verifier signature etc. Yuan *et al.* [14] in 2014 proposed a new certificateless threshold signature scheme (CLTHS) security model and a new certificateless threshold signature scheme based on bilinear mappings. Their security was based on a random oracle. Xiong *et al.* [15] in 2015 constructed a certificateless threshold signature scheme without random oracles or ideal ciphers for the first time. Gayathri *et al.* [13] in 2018 proposed a certificateless directed signature scheme without pairings in order to improve computing and communication efficiency.

In 2012, Zhang *et al.* [16] and Seo *et al.* [17] defined the security model of certificateless proxy signature scheme, and proposed the certificateless proxy signature schemes based on pairings, respectively. However, due to the computational complexity of pairings, He *et al.* [18] in 2013 proposed the first certificateless proxy signature scheme without pairings. This scheme can effectively improve efficiency and is especially suitable for practical applications where computational resources are severely constrained. Later, Lu and Li [19] in 2016 provided a certificateless proxy signature scheme with higher security, which can achieve unforgeability under adaptive chosen-message attacks and resist public key replacement attacks and a malicious KGC attack, and their scheme has relatively few public parameters and lower computational overheads. In addition, Du and Wen [20] presented the definition and security model of a certificateless proxy multiple signature (CLPMS) scheme based on pairings for the first time which was provably secure in the security model.

Boneh *et al.* [21] introduced the concept of aggregate signature in 2003. Subsequently, Xiong *et al.* [22] proposed an efficient certificateless aggregate signature (CAS) scheme with a constant pairing computation. This scheme is suitable for ad hoc networks, because it does not need synchronization to aggregate randomness. However, He *et al.* [23] in 2014 found that Xiong's scheme could not resist type II adversary who could forge any legal signature of messages. Later, Cheng *et al.* [24] analyzed Xiong et al.'s scheme and pointed out that Xiong et al.'s scheme could not resist the ''honesty but curiosity'' KGC attack. Then they put forward an improved scheme which could resist the ''malicious but passive'' KGC attack under the random oracle model. Subsequently, more CAS schemes are proposed and applied in practice. For example, Horng *et al.* [25] in 2015 and Kumar *et al.* [26] in 2018 proposed CAS schemes for vehicular sensor network and healthcare wireless sensor networks, respectively. Wireless sensors send detection data and signatures to cloud platforms for verification and verification, which will greatly benefit the development of human health care. In addition, Cui *et al.* [27] in 2018 and Zhong *et al.* [28] in 2019 respectively proposed CAS schemes in VANETs to reduce strong assumptions about ideal tamper-proof devices (TPD) as well as computing and communication costs. The difference between Cui's scheme and Zhong's scheme is that Cui's scheme does not use bilinear pairings, and their scheme is slightly more efficient than that of Zhong's.

In 2012, Tso *et al.* [11] proposed a strong secure certificateless short signature, which is more suitable for low bandwidth channels and/or low computing power systems requiring higher security levels. In 2018, Jia *et al.* [29] and Zhang *et al.* [30] put forward their certificateless signature schemes respectively. Besides, some medical application schemes were proposed. Ma *et al.* [31] in 2018 provided a certificateless signature scheme in the mobile medical system. And Shen *et al.* [32] proposed a lightweight online/offline certificateless signature scheme that can be applied to the medical system of the Internet of Things. It can realize the anonymous authentication of users in the wireless body area networks and low computational complexity and high efficiency. Certificateless signature is also used in cloud services and authentication protocols. Shen et al. in 2018 proposed a cloud-assisted lightweight certificateless authentication protocol [33], and a lightweight multi-layer authentication protocol for infinite body area network [34]. The former is anonymous, suitable for wireless body area network, and can be applied to telemedicine technology. The latter provides a certificateless protocol without pairings. For multi-group authentication protocols, the group key algorithm between personal digital assistant and each sensor node is established to achieve high energy efficiency and low computing cost. In addition, Zhou *et al.* [35] provided an effective certificateless signature scheme for cloud services. Miao *et al.* [36] proposed a verifiable multi-keyword search scheme (VMKS) based on certificateless cryptography, which realized the indistinguishability of ciphertexts and the unforgeability of signature. In other certificateless applications, a new trust anonymous authentication scheme for pervasive social networks was proposed by Yan et al. in 2018, which allows one or more authorized parties to publish the latest aggregate lists of integrated node trust for certificateless authenticating trust with unforgeability, nonlinkability and conditional traceability [37]. However, most of the above schemes are based on elliptic curve. In 2012, Zhang and Mao [38] proposed a certificateless signature scheme based on RSA, whose security is closely related to RSA and discrete logarithm problems.

And in the above signature schemes, most of them did not consider the undeniability of signatures. Duan [39] proposed the first certificateless undeniable signature scheme, which can realize the existential unforgeability for type I and type I adversaries. Moreover, both the confirmation protocol and the denial protocol can be proved by zero knowledge. Later, Zhao and Ye [40] proposed an efficient certificateless and undeniable signature scheme based on pairings. Compared with Duan's scheme, their scheme was more efficient than Duan's scheme, but they had a weakness, that is, they could only prove the unforgeability of their scheme in the weak model. Zhang *et al.* [41] pointed out that certificateless cryptography can solve the key escrow problem in IBC, but cannot solve the single point of failure. Therefore, they proposed a hierarchical certificateless signature scheme. Root KGC distributes workload by delegating part of private key generation and authentication to lower-level KGCs, which made multi-level KGCs and users form a tree structure. A user is a leaf node in the tree structure, so the verifier only needs to find the public key of the signer through the root KGC and the identity information of the signer.

Shim [42] compared the security models of Hu *et al.* [7], Huang *et al.* [10], and Au *et al.* [43]. Shim analyzed three certificateless signature schemes and put forward some suggestions for using hash function to realize valid signatures which can bind the signer's identity, public key, message and random value.

When the cryptosystem runs in real environment, it may be physically attacked by side channel attacks (SCA). In order to solve this problem, Huang *et al.* [44] in 2018 analysed the black-box construction of leakage-resilient ID-Based signature and certificateless signature. They defined the security models and proposed the structures of leakage-resilient identity-based signature and leakage-resilient certificateless signature based on leakage-resilient cryptography.

B. STRONG DESIGNATED VERIFIER SIGNATURE

The designated verifier signature (DVS) was first proposed by Jakobsson *et al.* [45] at Eurocrypt'96. In the designated verifier signature scheme, since the signature is shared by the signer and the designated verifier, the adversary can intercept the signature before the signature is sent and determine the signer.

In order to solve this problem, Saeednia *et al.* [46] proposed the concept of the strong designated verifier in 2003. The designated verifier generates a transcript that is indistinguishable from the original one, and only the designated verifier can verify the validity of the signature by using his own private key.

In order to solve the problems of designated verifier signature in traditional public key cryptosystem and ID-based cryptosystem, Huang *et al.* [47] proposed a certificateless designated verifier signature scheme. However, their scheme was insecure for ''malicious but passive'' KGC. In certificateless cryptography, the malicious but passive KGC can eavesdrop on partial private key sent to the user, but it cannot replace the user's public key and full private key.

Later, Yang *et al.* [48] in 2009 proposed an efficient certificateless strong designated verifier signature (CL-SDVS) scheme based on pairings. Later, Xiao *et al.* [49] in 2010 proposed a CL-SDVS scheme based on pairings. In 2011, Zhang et al. showed Xiao et al.'s scheme was insecure under a public key replacement attack and gave an improved scheme [50]. In 2012, Islam and Iswas [51] proposed a CL-SDVS scheme which is provably secure based on pairings. However, Liu *et al.* [52] in 2013 pointed out that in Islam et al.'s scheme, the private key generated by the user was involved in signature generation, but it did not play a role in signature verification, so the scheme could not resist malicious KGC attacks.

Afterwards, Chen *et al.* [53] in 2017 proposed a CL-SDVS scheme with non-delegation. In 2018, Lin [54] defined the first CL-SDVS scheme with signer ambiguity under key-compromise attacks and gave a new CL-SDVS scheme. The scheme not only has non-delegatability under adaptive chosen-message attacks, but also achieves signer ambiguity under key-compromise attacks.

In the strong designated verifier signatures, the original signature and the transcript are indistinguishable, which might easily induce that the signer and the designated verifier dispute the signature and deny their responsibility. In other words, these signature schemes are not undeniable. Hu *et al.* [55] in 2017 proposed an undeniable strong designated verifier signature schemes without pairings to solve this problem. If a trusted arbiter verifies a disputed signature, the signature which is a transcript generated by the designated verifier or the original signature generated by the signer could be easily identified. However, a signature which is neither the original signature nor the transcript will be mistakenly identified as the original signature signed by the signer. Subsequently, Hu *et al.* [56] proposed new undeniable strong designated verifier signature schemes which solved the above problem. Arbiter can identify whether the disputed signature is the original signature or a transcript, or neither of them.

III. PRELIMINARIES

In this section, we will review the mathematical knowledge of elliptic curves and some difficult problems.

A. ELLIPTIC CURVE

Let p be a large prime number and $GF(p)$ be a prime finite field. All points over an elliptic curve E/F_p along with a special point at infinity O form a cyclic additive group G_1 of order n. P is a generator of group G_1 if n is the smallest number such that $nP = O$.

Here are some rules on elliptic curves:

1) The point O is additive identity of the group G_1.
2) A line perpendicular to the X-axis intersects the elliptic curve at two points, the X coordinates of which are equal, that is, $P_1 = (x, y)$ and $P_2 = (x, -y)$. It intersects with the curve at infinity O, so $P_1 = -P_1$.
3) All points $P = (x, y) \in E_p(a, b)$ on an elliptic curve, all points satisfy $P + O = O + P = P$.
4) There are $P = (x_1, y_1) \in E_p(a, b)$ and $Q = (x_2, y_2) \in E_p(a, b)$, then $P + Q = (x_3, y_3) \in E_p(a, b)$ which satisfies: $x_3 = \lambda^2 - x_1 - x_2$, $y_3 = \lambda(x_1 - x_3) - y_1$ and $\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} if P \neq Q \\ \frac{3x_1^2 + a}{2y_1} if P = Q. \end{cases}$
5) There are $P, Q \in E_p(a, b)$, then $P + Q = Q + P$.
6) There are $P, Q, R \in E_p(a, b)$, then $P + (Q + R) = (P + Q) + R$.
7) If k is an integer, for all points $P \in E_p(a, b)$ satisfies: $k \cdot P = P + P + P + \cdots + P$ (there are kP additions).
8) *If s and t are integers, for all points $P \in E_p(a, b)$ satisfies:* $(s + t) \cdot P = s \cdot P + t \cdot P$ *and* $s \cdot (t \cdot P) = t \cdot (s \cdot P)$.

B. COMPUTATIONAL ASSUMPTION

1) COMPUTATIONAL DIFFIE-HELLMAN PROBLEM (CDHP)

Given $P, aP, bP \in G_1$ where G_1 is a cyclic group of order n, $P \in G_1$ and $a, b \in Z_q^*$, and it is computationally intractable to compute the value $abP \in G_1$.

2) COMPUTATIONAL DIFFIE-HELLMAN (CDH) ASSUMPTION

The advantage for every probabilistic polynomial-time algorithm $\mathcal{A}$ to solve the CDH is negligible.

Definition 1: The (t, ε)-CDH assumption holds if there is no polynomial-time algorithm $\mathcal{A}$ that can solve the CDHP in time at most t and with an advantage ε.

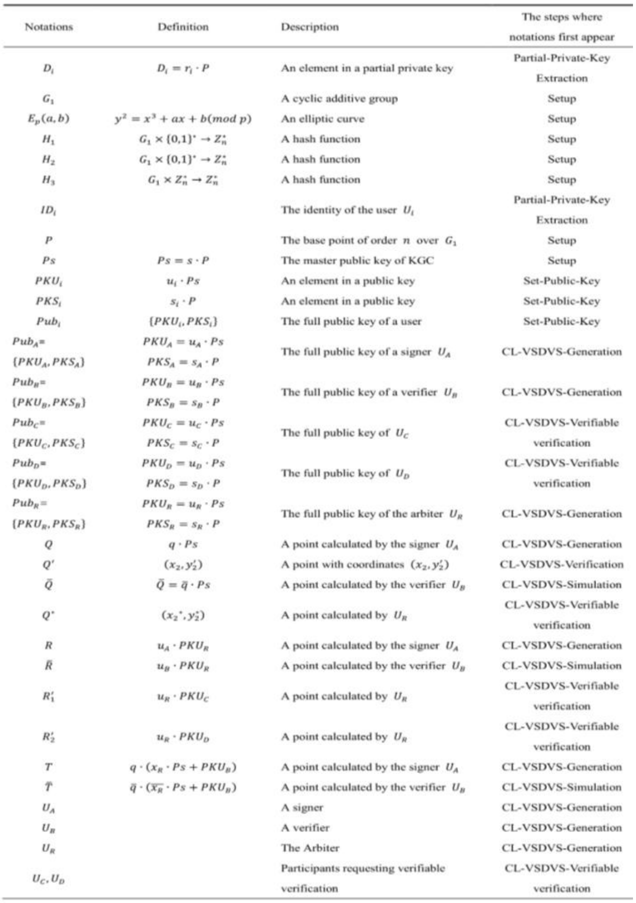
3) ELLIPTIC CURVE DISCRETE LOGARITHM PROBLEM (ECDLP)

Given a point $Q = aP \in G_1$ where $a \in Z_n$, it is infeasible to compute a with non-negligible probability.

4) ELLIPTIC CURVE DISCRETE LOGARITHM (ECDL) ASSUMPTION

The advantage for every probabilistic polynomial-time algorithm $\mathcal{A}$ to solve the ECDLP is negligible.

TABLE 1. Displays the symbols used in the next section.



Notations	Definition	Description	The steps where notations first appear
D_i	$D_i = r_i \cdot P$	An element in a partial private key	Partial-Private-Key Extraction
G_1		A cyclic additive group	Setup
$E_p(a,b)$	$y^2 = x^3 + ax + b(mod\ p)$	An elliptic curve	Setup
H_1	$G_1 \times \{0,1\}^* \rightarrow Z_n^*$	A hash function	Setup
H_2	$G_1 \times \{0,1\}^* \rightarrow Z_n^*$	A hash function	Setup
H_3	$G_1 \times Z_n^* \rightarrow Z_n^*$	A hash function	Setup
ID_i		The identity of the user U_i	Partial-Private-Key Extraction
P		The base point of order n over G_1	Setup
Ps	$Ps = s \cdot P$	The master public key of KGC	Setup
PKU_i	$u_i \cdot Ps$	An element in a public key	Set-Public-Key
PKS_i	$s_i \cdot P$	An element in a public key	Set-Public-Key
Pub_i	$\{PKU_i, PKS_i\}$	The full public key of a user	Set-Public-Key
Pub_A= $\{PKU_A, PKS_A\}$	$PKU_A = u_A \cdot Ps$ $PKS_A = s_A \cdot P$	The full public key of a signer U_A	CL-VSDVS-Generation
Pub_B= $\{PKU_B, PKS_B\}$	$PKU_B = u_B \cdot Ps$ $PKS_B = s_B \cdot P$	The full public key of a verifier U_B	CL-VSDVS-Generation
Pub_C= $\{PKU_C, PKS_C\}$	$PKU_C = u_C \cdot Ps$ $PKS_C = s_C \cdot P$	The full public key of U_C	CL-VSDVS-Verifiable verification
Pub_D= $\{PKU_D, PKS_D\}$	$PKU_D = u_D \cdot Ps$ $PKS_D = s_D \cdot P$	The full public key of U_D	CL-VSDVS-Verifiable verification
Pub_R= $\{PKU_R, PKS_R\}$	$PKU_R = u_R \cdot Ps$ $PKS_R = s_R \cdot P$	The full public key of the arbiter U_R	CL-VSDVS-Generation
Q	$q \cdot Ps$	A point calculated by the signer U_A	CL-VSDVS-Generation
Q'	(x_2, y_2')	A point with coordinates (x_2, y_2')	CL-VSDVS-Verification
$\bar{Q}$	$\bar{Q} = \bar{q} \cdot Ps$	A point calculated by the verifier U_B	CL-VSDVS-Simulation
Q^*	$(x_2{}^*, y_2^*)$	A point calculated by U_R	CL-VSDVS-Verifiable verification
R	$u_A \cdot PKU_R$	A point calculated by the signer U_A	CL-VSDVS-Generation
$\bar{R}$	$u_B \cdot PKU_R$	A point calculated by the verifier U_B	CL-VSDVS-Simulation
R_1'	$u_R \cdot PKU_C$	A point calculated by U_R	CL-VSDVS-Verifiable verification
R_2'	$u_R \cdot PKU_D$	A point calculated by U_R	CL-VSDVS-Verifiable verification
T	$q \cdot (x_R \cdot Ps + PKU_B)$	A point calculated by the signer U_A	CL-VSDVS-Generation
$\bar{T}$	$\bar{q} \cdot (\overline{x_R} \cdot Ps + PKU_B)$	A point calculated by the verifier U_B	CL-VSDVS-Simulation
U_A		A signer	CL-VSDVS-Generation
U_B		A verifier	CL-VSDVS-Generation
U_R		The Arbiter	CL-VSDVS-Generation
U_C, U_D		Participants requesting verifiable verification	CL-VSDVS-Verifiable verification

Definition 2: The (t, ε)- ECDL assumption holds if there is no polynomial-time algorithm $\mathcal{A}$ that can solve the ECDLP in time at most t and with an advantage ε.

IV. SYSTEM MODEL AND SECURITY MODEL

We will provide the system model and security model for certificateless verifiable designated verifier signatures.

A. SYSTEM MODEL

The system model in Figure 1 consists of four participants: the third-party key generation center (KGC), a signer, a verifier and the secure and trusted arbiter. KGC is used to set up the system and provides partial private keys of users (signers, verifiers and the arbiter). KGC has its own master private key and public key. The signer, the verifier and the arbiter obtain partial private keys provided by KGC and secret values selected by itself to generate the full private keys and public keys, respectively. In the signature generation, the signer uses the public keys of the verifier and the arbiter as well as the random number selected by itself to generate the corresponding signature and sends it to the verifier. In the signature verification, the verifier uses public keys of the signer and private keys of itself to verify the validity of the signature. After signature verification is successful, the verifier generates a transcript of the signature by using public keys of the signer, the arbiter and a random integer selected by itself. When the signer and verifier dispute the signature, either party can send the disputed signature to the arbiter, and request the verifiable verification. The other party uses public keys of the arbiter, the signer and its own private key to generate parameters which will be sent to the arbiter. The arbiter indirectly verifies the validity of the signature and the correctness of the verification result by using its private key and the information sent by both parties. Finally, the arbiter returns the result to the signer and the verifier.

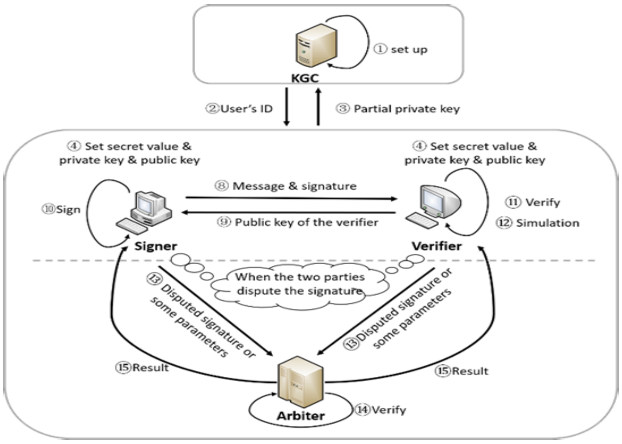

KGC
① set up
②User's ID
③ Partial private key
④ Set secret value & private key & public key
④ Set secret value & private key & public key
⑧ Message & signature
⑨ Public key of the verifier
⑩Sign
⑪ Verify
⑫ Simulation
Signer
Verifier
When the two parties dispute the signature
⑬ Disputed signature or some parameters
⑬Disputed signature or some parameters
⑮Result
⑮Result
Arbiter
⑭Verify

FIGURE 1. System model.

B. ADVERSARY TYPES

The security models for a certificateless strong designated verifier signature scheme are defined by the follow two games for Type I adversary and Type II adversary, respectively. Each of the games will capture the notion of existential unforgeability under adaptive chosen-message attacks. Type I adversary is a malicious user who cannot access the master key and the target user's partial private key, but can replace the public key of any user. Type II adversary is a malicious but passive KGC [44], who can obtain access the master key and the user's partial private key, but cannot get the user's secret value key nor replace the user's public key.

Next, we will define three games to verify the security of our scheme.

C. SECURITY MODELS

This subsection gives security games which assume what the adversary could do against the proposed scheme.

GAME_1: The game between challenger $\mathcal{C}$ and adversary $\mathcal{A}$ are used to simulate the adaptive chosen-message attacks of a Type I adversary.

We use the following challenger-adversary interactive games to simulate the adaptive chosen-message attacks of the Type I adversary.

Setup: The challenger $\mathcal{C}$ inputs a security parameter l, runs the Setup algorithm, and generates the master-private key, master-public key and other system parameters. The challenger $\mathcal{C}$ returns the public system parameters to the adversary $\mathcal{A}$.

Hash-Query: The adversary $\mathcal{A}$ can send any input to the challenger $\mathcal{C}$, and the challenger returns the corresponding hash output to the adversary.

Create-Users: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger $\mathcal{C}$, and the challenger runs the Partial-Private-Key Extraction algorithm, the Set-Secret-Value algorithm, the Set-Private-Key algorithm and the Set-Public-Key algorithm and updates *Partial_private_key_list*, *Secret_value_list*, *Public_key_list*. Finally, the challenger returns the corresponding public key $Pub_i = \{PKU_i, PKS_i\}$ to the adversary.

Finally, $\mathcal{C}$ adds $(ID_i, s_i, u_i, PKU_i, PKS_i)$ to *user_list*.

If the ID already exists in the *user_list*, the corresponding public key is returned directly.

Partial-Private-key-Query: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ first searches partial-private-key-list for the corresponding *ID*. If it does not exist, the challenger runs Create-Users algorithm and updates the corresponding list.

And finally returns the corresponding partial private key to the adversary. Conversely, the challenger returns the search results to the adversary.

Secret-Value-Query: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger. If it does not exist, the challenger runs Create-Users algorithm and updates the corresponding list. Finally, the challenger returns the corresponding secret value to the adversary. Conversely, the challenger returns the search results to the adversary.

Public-key-Query: The adversary $\mathcal{A}$ inputs ID_i to request Public-key-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs the Create-Users algorithm to obtain the public key $\{PKU_i, PKS_i\}$. Finally, $\mathcal{C}$ returns the public key $\{PKU_i, PKS_i\}$ to $\mathcal{A}$ and adds (ID_i, PKU_i, PKS_i) to *Public_key_list*.

If $\mathcal{C}$ finds the corresponding record in the *Public_key_list*, then returns $\{PKU_i, PKS_i\}$ to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

Replace-Public-key-Query: The adversary $\mathcal{A}$ sends a user's *ID* and a new public key $Pub'_i = \left\{PKU'_i, PKS'_i\right\}$ to the challenger $\mathcal{C}$. The Challenger updates the list of users' public keys and replaces them with the corresponding users' public keys.

CL-VSDVS-Generation-Query: The adversary $\mathcal{A}$ sends the signer's ID_i, verifier's ID_j and message m to the challenger $\mathcal{C}$. The Challenger runs the CL-VSDVS-Generation algorithm and sends the corresponding signature σ to the adversary.

CL-VSDVS-Verification-Query: The adversary $\mathcal{A}$ sends the signer's ID_i, verifier's ID_j and message m and the corresponding signature σ to the challenger $\mathcal{C}$. The challenger runs the CL-VSDVS-Verification algorithm and returns the result of the signature verification to the adversary.

Forgery: Finally, the adversary $\mathcal{A}$ gives a quaternion $\left(m^*, \sigma^*, ID_i^*, ID_j^*\right)$ which satisfy:

① The signature σ^* is the valid signature of ID_i^*, ID_j^* and the message m^*. The adversary $\mathcal{A}$ has never submitted the CL-VSDVS-Generation-query of ID_i^*, ID_j^* and m^* to the challenger $\mathcal{C}$.

② The adversary $\mathcal{A}$ has never submitted Partial-private-key-query and Secret-value-query about ID_i^*, ID_j^* to challenger $\mathcal{C}$.

GAME_2: We use the following challenger-adversary interactive games to simulate the adaptive chosen-message attacks of the Type II adversary.

Setup: The challenger $\mathcal{C}$ inputs a security parameter l, runs the Setup algorithm, and generates the master-private key, master-public key and other system parameters. The challenger $\mathcal{C}$ returns the master-private key and public system parameters to the adversary $\mathcal{A}$.

Hash-Query: The adversary $\mathcal{A}$ can send any input to the challenger $\mathcal{C}$, and the challenger returns the corresponding hash output to the adversary.

Create-Users: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger $\mathcal{C}$, and the challenger runs the Partial-Private-Key Extraction algorithm, the Set-Secret-Value algorithm, the Set-Private-Key algorithm and the Set-Public-Key algorithm and updates *Partial_private_key_list*, *Secret_value_list*, *Public_key_list*. Finally, the challenger returns the corresponding public key $Pub_i = \{PKU_i, PKS_i\}$ to the adversary.

Finally, $\mathcal{C}$ adds $(ID_i, s_i, u_i, PKU_i, PKS_i)$ to *user_list*.

If the ID already exists in the *user_list*, the corresponding public key is returned directly.

Partial-Private-key-Query: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs Partial-Private-Key Extraction algorithm, and finally returns the corresponding partial private key to the adversary.

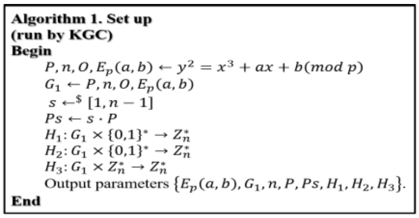
Secret-Value-Query: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger. The challenger $\mathcal{C}$ runs the Set-Secret-Value algorithm and returns the corresponding secret value to the adversary.

Public-key-Query: The adversary $\mathcal{A}$ sends a user's *ID* to the challenger. The challenger $\mathcal{C}$ runs the Set-Public-Key algorithm and returns the corresponding public key to the adversary.

CL-VSDVS-Generation-Query: The adversary $\mathcal{A}$ sends the signer's ID_i, verifier's ID_j and message m to the challenger $\mathcal{C}$. The Challenger runs the CL-VSDVS-Generation algorithm and sends the corresponding signature σ to the adversary.

CL-VSDVS-Verification-Query: The adversary $\mathcal{A}$ sends the signer's ID_i, verifier's ID_j and message m and the corresponding signature σ to the challenger $\mathcal{C}$. The challenger runs the CL-VSDVS-Verification algorithm and returns the result of the signature verification to the adversary.

Forgery: The adversary $\mathcal{A}$ gives a quaternion $\left(m^*, \sigma^*, ID_i^*, ID_j^*\right)$ which satisfy:



```
Algorithm 1. Set up
(run by KGC)
Begin
    $P, n, O, E_p(a,b) \leftarrow y^2 = x^3 + ax + b(mod\ p)$
    $G_1 \leftarrow P, n, O, E_p(a,b)$
    $s \leftarrow^{\$} [1, n-1]$
    $Ps \leftarrow s \cdot P$
    $H_1: G_1 \times \{0,1\}^* \rightarrow Z_n^*$
    $H_2: G_1 \times \{0,1\}^* \rightarrow Z_n^*$
    $H_3: G_1 \times Z_n^* \rightarrow Z_n^*$
    Output parameters $\{E_p(a,b), G_1, n, P, Ps, H_1, H_2, H_3\}$.
End
```

FIGURE 2. Algorithm 1.

① The signature σ^* is the valid signature of ID_i^*, ID_j^* and the message m^*. The adversary $\mathcal{A}$ has never submitted the CL-VSDVS-Generation-query of ID_i^*, ID_j^* and m^* to the challenger $\mathcal{C}$.

② The adversary $\mathcal{A}$ has never submitted and Secret-value-query about ID_i^*, ID_j^* to challenger $\mathcal{C}$.

Definition 3: In the random oracle model, if there is no adversary having non-negligible advantage to solve ECDLP, then a certificateless verifiable strong designated verifier signature scheme satisfies the security requirements of existentially unforgeable under adaptive chosen-message attacks.

Definition 4: Verifiability: A certificateless verifiable strong designated verifier signature scheme satisfies verifiability if it is not computationally feasible to determine who is the signer or verifier except that the arbiter can judge whether the signature generated by the signer or the verifier when there is a dispute between the signer and the verifier.

Definition 5: Non-transferability: A certificateless verifiable strong designated verifier signature scheme satisfies non-transferability if it is not computationally feasible to determine who is the signer or verifier.

V. PROPOSED SCHEME

This section describes our proposed CL-VSDVS scheme. Here we assume that the signer is U_A, the designated verifier is U_B, and the arbitrator is U_R.

Setup: On inputting a security parameter l, the KGC selects a large prime number p and an elliptic curve $E_p(a,b) : y^2 = x^3 + ax + b\,(mod\ p)$ over filed $GF(p)$ satisfying that $4a^3 + 27b^2\,(mod\ p) \neq 0$ where $a, b \epsilon F_p$. Let G_1 be a cyclic additive group consisting of the points on $E_p(a,b)$ and a special point at infinity O. P is a base point of order n over G_1. KGC randomly picks $s \in [1, n-1]$ as the master private key and calculates $Ps = s \cdot P$ as the master public key. KGC also picks three secure hash functions: $H_1 : G_1 \times \{0,1\}^* \rightarrow Z_n^*$, $H_2 : G_1 \times \{0,1\}^* \rightarrow Z_n^*$, $H_3 : G_1 \times Z_n^* \rightarrow Z_n^*$. The public parameters are composed of $\left\{E_p(a,b), G_1, n, P, Ps, H_1, H_2, H_3\right\}$.

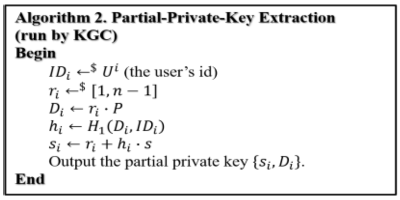
The algorithm is shown in Figure 2 and step ① in Figure 7.

Partial-Private-Key Extraction: Each user U_i can request his/her partial private key from the KGC by inputting an identity ID_i. KGC randomly chooses $r_i \in [1, n-1]$ to compute:

$$\mathrm{D_i} = \mathrm{r_i} \cdot \mathrm{P}, \tag{1}$$

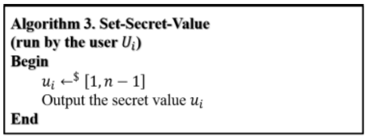
$$h_i = H_1(D_i, ID_i), \tag{2}$$

$$s_i = r_i + h_i s. \tag{3}$$



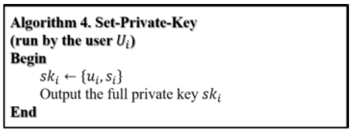
```
Algorithm 2. Partial-Private-Key Extraction
(run by KGC)
Begin
    $ID_i \leftarrow^{\$} U^i$ (the user's id)
    $r_i \leftarrow^{\$} [1, n-1]$
    $D_i \leftarrow r_i \cdot P$
    $h_i \leftarrow H_1(D_i, ID_i)$
    $s_i \leftarrow r_i + h_i \cdot s$
    Output the partial private key $\{s_i, D_i\}$.
End
```

FIGURE 3. Algorithm 2.



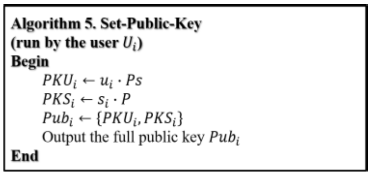
```
Algorithm 3. Set-Secret-Value
(run by the user $U_i$)
Begin
    $u_i \leftarrow^{\$} [1, n-1]$
    Output the secret value $u_i$
End
```

FIGURE 4. Algorithm 3.



```
Algorithm 4. Set-Private-Key
(run by the user $U_i$)
Begin
    $sk_i \leftarrow \{u_i, s_i\}$
    Output the full private key $sk_i$
End
```

FIGURE 5. Algorithm 4.



```
Algorithm 5. Set-Public-Key
(run by the user $U_i$)
Begin
    $PKU_i \leftarrow u_i \cdot Ps$
    $PKS_i \leftarrow s_i \cdot P$
    $Pub_i \leftarrow \{PKU_i, PKS_i\}$
    Output the full public key $Pub_i$
End
```

FIGURE 6. Algorithm 5.

Then KGC sends the partial private key $\{s_i, D_i\}$ to the user via a secure channel. The user U_i can compute the $h_i = H_1(D_i, ID_i)$ to check the validity of the key. The correctness is guaranteed by the following equation:

$$s_i \cdot P = D_i + h_i \cdot Ps. \tag{4}$$

The algorithm is shown in Figure 3 and steps ② and ③ in Figure 7.

Set-Secret-Value: The user U_i randomly selects an integer number $u_i \in [1, n-1]$ and sets u_i as his/her secret value.

The algorithm is shown in Figure 4 and step ④ in Figure 7.

Set-Private-Key: The user U_i sets $sk_i = \{u_i, s_i\}$ as his full private key.

The algorithm is shown in Figure 5 and step ⑤ in Figure 7.

Set-Public-Key: The user U_i computes:

$$PKU_i = u_i \cdot Ps, \tag{5}$$

$$PKS_i = s_i \cdot P. \tag{6}$$

Then user U_i sets $Pub_i = \{PKU_i, PKS_i\}$ as his/her full public key.

The algorithm is shown in Figure 6 and step ⑥ in Figure 7.

CL-VSDVS-Generation: Before the signature generation, the signer calculates $R = u_A \cdot PKU_R$ with the trusted

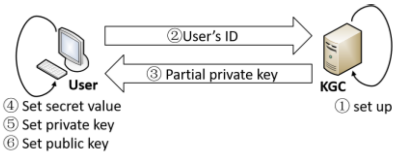
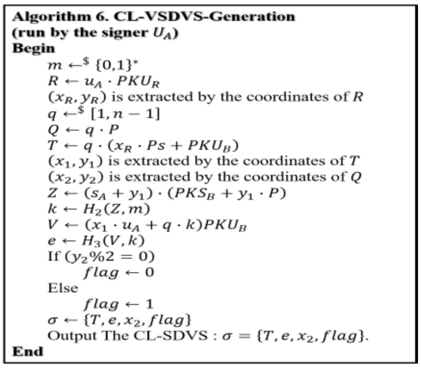


FIGURE 7. Setup and key generation of the proposed scheme.



```
Algorithm 6. CL-VSDVS-Generation
(run by the signer $U_A$)
Begin
    $m \leftarrow^{\$} \{0,1\}^*$
    $R \leftarrow u_A \cdot PKU_R$
    $(x_R, y_R)$ is extracted by the coordinates of $R$
    $q \leftarrow^{\$} [1, n-1]$
    $Q \leftarrow q \cdot P$
    $T \leftarrow q \cdot (x_R \cdot Ps + PKU_B)$
    $(x_1, y_1)$ is extracted by the coordinates of $T$
    $(x_2, y_2)$ is extracted by the coordinates of $Q$
    $Z \leftarrow (s_A + y_1) \cdot (PKS_B + y_1 \cdot P)$
    $k \leftarrow H_2(Z, m)$
    $V \leftarrow (x_1 \cdot u_A + q \cdot k) PKU_B$
    $e \leftarrow H_3(V, k)$
    If $(y_2\%2 = 0)$
        $flag \leftarrow 0$
    Else
        $flag \leftarrow 1$
    $\sigma \leftarrow \{T, e, x_2, flag\}$
    Output The CL-SDVS : $\sigma = \{T, e, x_2, flag\}$.
End
```

FIGURE 8. Algorithm 6.

arbiter U_R 's public key. And (x_R, y_R) is extracted by the coordinates of R.

For generating a signature for message $m \in \{0,1\}^*$ intended for the user U_B, the signer U_A randomly selects $q \in [1, n-1]$ to compute:

$$Q = q \cdot Ps \tag{7}$$

$$T = q \cdot (x_R \cdot Ps + PKU_B), \tag{8}$$

$$Z = (s_A + y_1) \cdot (PKS_B + y_1 \cdot P), \tag{9}$$

$$k = H_2(Z, m), \tag{10}$$

$$V = (x_1 \cdot u_A + q \cdot k) \cdot PKU_B, \tag{11}$$

$$e = H_3(V, k). \tag{12}$$

where (x_1, y_1) is extracted by the coordinates of T and (x_2, y_2) is extracted by the coordinates of Q. If y_2 is even, set the *flag* with a length of 1 bit to 0, otherwise, set the *flag* to 1. The CL-VSDVS on message m is $\sigma = \{T, e, x_2, flag\}$ which will be sent to the designated verifier U_B.

The algorithm is shown in Figure 8 and steps ①, ② and ③ in Figure 10.

CL-VSDVS-Verification: The designated verifier U_B received $\sigma = \{T, e, x_2, flag\}$, and verify its validity. In order to verify it, the designated verifier U_B uses x_2 to calculate y_2' according to the elliptic curve formula. Since the y_2' obtained by the calculation has two values, the true value of y_2' is determined according to the *flag*. The designated verifier U_B sets $Q' = (x_2, y_2')$. Then the verifier computes:

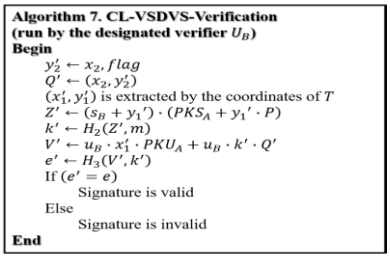
$$Z' = (s_B + y_1') \cdot (PKS_A + y_1' \cdot P), \tag{13}$$

$$k' = H_2(Z', m), \tag{14}$$

$$V' = u_B \cdot x_1' \cdot PKU_A + u_B \cdot k' \cdot Q', \tag{15}$$

$$e' = H_3(V', k') \tag{16}$$

where (x_1', y_1') is extracted by the coordinates of T.



```
Algorithm 7. CL-VSDVS-Verification
(run by the designated verifier $U_B$)
Begin
    $y_2' \leftarrow x_2, flag$
    $Q' \leftarrow (x_2, y_2')$
    $(x_1', y_1')$ is extracted by the coordinates of $T$
    $Z' \leftarrow (s_B + y_1') \cdot (PKS_A + y_1' \cdot P)$
    $k' \leftarrow H_2(Z', m)$
    $V' \leftarrow u_B \cdot x_1' \cdot PKU_A + u_B \cdot k' \cdot Q'$
    $e' \leftarrow H_3(V', k')$
    If $(e' = e)$
        Signature is valid
    Else
        Signature is invalid
End
```

FIGURE 9. Algorithm 7.

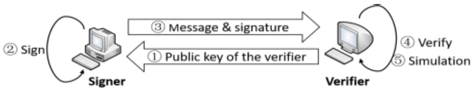


FIGURE 10. Signature generation, verification and simulation of the proposed scheme.

If $e' = e$, the designated verifier U_B is convinced of the validity of $\sigma = \{T, e, x_2, flag\}$. The correctness of the scheme is guaranteed by the following description.

The algorithm is shown in Figure 9 and steps ③ and ④ in Figure 10.

A. CORRECTNESS DESCRIPTION

Then we can know that

$$\begin{aligned} Z' &= \left(s_B + y_1'\right) \cdot \left(PKS_A + y_1' \cdot P\right) \\ &= \left(s_B + y_1'\right) \cdot \left(s_A + y_1'\right) \cdot P \\ &= \left(s_A + y_1'\right) \cdot \left(PKS_B + y_1' \cdot P\right) = Z \\ k' &= H_2\left(Z', m\right) \\ &= H_2(Z, m) = k \\ V' &= \left(u_B \cdot x_1' \cdot PKU_A + u_B \cdot k' \cdot Q\right) \\ &= u_B \cdot \left(x_1' \cdot u_A + k' \cdot q\right) \cdot Ps \\ &= \left(x_1' \cdot sk_A + q \cdot k\right) PKU_B \end{aligned}$$

by the equation (7) (9) (10) and (11). Finally, we can obtain that $e' = H_3\left(V', k'\right) = H_3(V, k) = e$ by the equation (12).

CL-VSDVS-Simulation: The designated verifier U_B calculates $\overline{R} = u_B \cdot PKU_R$ with the trusted arbiter U_R 's public key. And $(\overline{x_R}, \overline{y_R})$ is extracted by the coordinates of $\overline{R}$.

selects randomly $\overline{q} \in [1, n-1]$, then U_B computes:

$$\overline{Q} = \overline{q} \cdot Ps, \tag{17}$$

$$\overline{T} = \overline{q} \cdot (\overline{x_R} \cdot Ps + PKU_B), \tag{18}$$

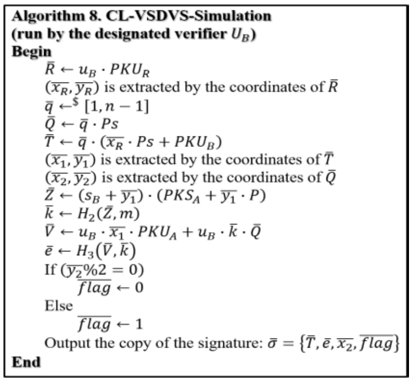
$$\overline{Z} = (s_B + \overline{y_1}) \cdot (PKS_A + \overline{y_1} \cdot P), \tag{19}$$

$$\overline{k} = H_2\left(\overline{Z}, m\right), \tag{20}$$

$$\overline{V} = u_B \cdot \overline{x_1} \cdot PKU_A + u_B \overline{k} \cdot \overline{Q}, \tag{21}$$

$$\overline{e} = H_3\left(\overline{V}, \overline{k}\right). \tag{22}$$

where $(\overline{x_1}, \overline{y_1})$ is extracted by the coordinates of $\overline{T}$ and $(\overline{x_2}, \overline{y_2})$ is extracted by the coordinates of $\overline{Q}$. If $\overline{y_2}$ is even, set the *flag* with a length of 1 bit to 0, otherwise, set the *flag* to 1.



```
Algorithm 8. CL-VSDVS-Simulation
(run by the designated verifier $U_B$)
Begin
    $\bar{R} \leftarrow u_B \cdot PKU_R$
    $(\overline{x_R}, \overline{y_R})$ is extracted by the coordinates of $\bar{R}$
    $\bar{q} \leftarrow^{\$} [1, n-1]$
    $\bar{Q} \leftarrow \bar{q} \cdot Ps$
    $\bar{T} \leftarrow \bar{q} \cdot (\overline{x_R} \cdot Ps + PKU_B)$
    $(\overline{x_1}, \overline{y_1})$ is extracted by the coordinates of $\bar{T}$
    $(\overline{x_2}, \overline{y_2})$ is extracted by the coordinates of $\bar{Q}$
    $\bar{Z} \leftarrow (s_B + \overline{y_1}) \cdot (PKS_A + \overline{y_1} \cdot P)$
    $\bar{k} \leftarrow H_2(\bar{Z}, m)$
    $\bar{V} \leftarrow u_B \cdot \overline{x_1} \cdot PKU_A + u_B \cdot \bar{k} \cdot \bar{Q}$
    $\bar{e} \leftarrow H_3(\bar{V}, \bar{k})$
    If $(\overline{y_2}\%2 = 0)$
        $\overline{flag} \leftarrow 0$
    Else
        $\overline{flag} \leftarrow 1$
    Output the copy of the signature: $\bar{\sigma} = \{\bar{T}, \bar{e}, \overline{x_2}, \overline{flag}\}$
End
```

FIGURE 11. Algorithm 8.

Obviously, the transcript $\overline{\sigma} = \left\{\overline{T}, \overline{e}, \overline{x_2}, \overline{flag}\right\}$ on the message m also can also be verified.

The algorithm is shown in Figure 11 and step ⑤ in Figure 10.

CL-VSDVS-Verifiable Verification: When the signer U_A and verifier U_B dispute signature $\sigma^* = \left\{T^*, e^*, x_2^*, flag^*\right\}$ of the message m^*.

The results of signature verification are as follows: the first case is that both signer and verifier think the signature is valid; the second case is that both signer and verifier think that the signature is invalid; the third case is that signer insists the signature is valid but the verifier insists the signature is invalid; the fourth case is that signer insists the signature is invalid but the verifier insists the signature is valid. In the first and second cases, there is no dispute between the signer and the verifier; in the third and fourth cases, there is dispute between the signer and the verifier. And the above verification results can be verified by the arbiter.

When a dispute arises between the signer and the verifier, we assume that the participant who considers signature is invalid is U_C, and the participant who considers signature is valid is U_D.

U_C as a claimant will send the disputed signature to the trusted arbiter U_R, who will receive the signature and send it to U_D for confirmation whether the signature sent by U_R is the disputed one. After the confirmation is successful, U_D as a prover computes $Y = s_D \cdot PKS_R$ and gets $Y_1 = x_D \cdot s_D \cdot PKS_C$ and $Y_2 = x_D \cdot u_D \cdot PKU_C$ where x_D is the x coordinate of Y, then sends $\{Y_1, Y_2\}$ to the arbiter.

After the arbiter U_R receives the signature and verification parameters $\{Y_1, Y_2\}$, U_R computes $R_1' = u_R \cdot PKU_C$ and $R_2' = u_R \cdot PKU_D$, and $\left(x_{R1}', y_{R1}'\right)$ and $\left(x_{R2}', y_{R2}'\right)$ are extracted by the coordinates of R_1' and R_2' respectively. The arbiter U_R uses x_2^* to calculate y_2^* according to the elliptic curve formula. Since the y_2^* obtained by the calculation has two values, the true value of y_2^* is determined according to the $flag^*$. The arbiter U_R sets $Q^* = \left(x_2^*, y_2^*\right)$ and computes $Y' = s_R \cdot PKS_D$, and gets $Y_1' = \left(x_D'\right)^{-1} \cdot Y_1$ and $Y_2' = \left(x_D'\right)^{-1} \cdot Y_2$ where x_D' is the x coordinate of Y'.

Finally, the arbiter U_R computes:

$$\begin{aligned}
Z_1^* &= Y_1' + y_1^* \cdot (PKS_C + PKS_D) + y_1^* \cdot y_1^* \cdot P,\\
k_1^* &= H_2\left(Z_1^*, m^*\right),\\
V_1^* &= x_1^* \cdot Y_2' + k^* \cdot \left(T^* - x_{R1}' \cdot Q^*\right),\\
V_2^* &= x_1^* \cdot Y_2' + k^* \cdot \left(T^* - x_{R2}' \cdot Q^*\right),\\
e_1^* &= H_3\left(V_1^*, k^*\right),\\
e_2^* &= H_3\left(V_2^*, k^*\right).
\end{aligned}$$

where $\left(x_1^*, y_1^*\right)$ is extracted by the coordinates of T^*.

If $e_1^* = e^*$, then the arbiter judges that U_D is right and the signature is valid and produced by U_C. Else if $e_2^* = e^*$, then the arbiter judges that the signature is valid and produced by U_D. When $e_1^* \neq e^*$ and $e_2^* \neq e^*$, the arbiter judges that the signature is invalid and U_C is right.

The algorithm is shown in Figure 3 and Figure 2.

B. CORRECTNESS DESCRIPTION

Here, we use the following case to illustrate the correctness. We assume that U_C is the signer and U_D is the verifier and the signature is valid and generated by U_C. And according to equation (7) and (8), we can get that

$$\begin{aligned}
T^* - x_R' \cdot Q^* &= q \cdot (x_R \cdot PS + PKU_D) - x_{R1}' \cdot q \cdot Ps\\
&= q{\cdot}PKU_D = u_D \cdot Q^*.
\end{aligned}$$

And we can get $Y_1' = \left(x_D'\right)^{-1} \cdot Y_1 = s_C \cdot s_D \cdot P$ and $Y_2' = \left(x_D'\right)^{-1} \cdot Y_2 = u_C \cdot u_D \cdot P$.

Therefore, we can compute:

$$\begin{aligned}
Z_1^* &= Y_1' + y_1^* \cdot (PKS_C + PKS_D) + y_1^* \cdot y_1^* \cdot P\\
&= s_C \cdot s_D \cdot P + y_1^* \cdot (s_C \cdot P + s_D \cdot P) + y_1^* \cdot y_1^* \cdot P\\
&= s_C \cdot \left(y_1^* + s_D\right) \cdot P + y_1^* \cdot \left(s_D + y_1^*\right) \cdot P\\
&= \left(y_1^* + s_C\right) \cdot \left(y_1^* + s_D\right) \cdot P\\
&= \left(y_1^* + s_C\right) \cdot \left(y_1^* \cdot P + PKS_D\right) = Z\\
k^* &= H_2\left(Z_1^*, m^*\right) = k\\
V_1^* &= x_1^* \cdot Y_2' + k^* \cdot \left(T^* - x_R' \cdot Q^*\right)\\
&= x_1^* \cdot u_D \cdot PKU_C + k^* \cdot u_D \cdot Q^*\\
&= \left(x_1^* \cdot u_C + q \cdot k\right) {\cdot} PKU_D = V
\end{aligned}$$

Finally, we get the conclusion: $e_1^* = H_3\left(V^*, k^*\right) = e^*$.

VI. SECURITY ANALYSIS

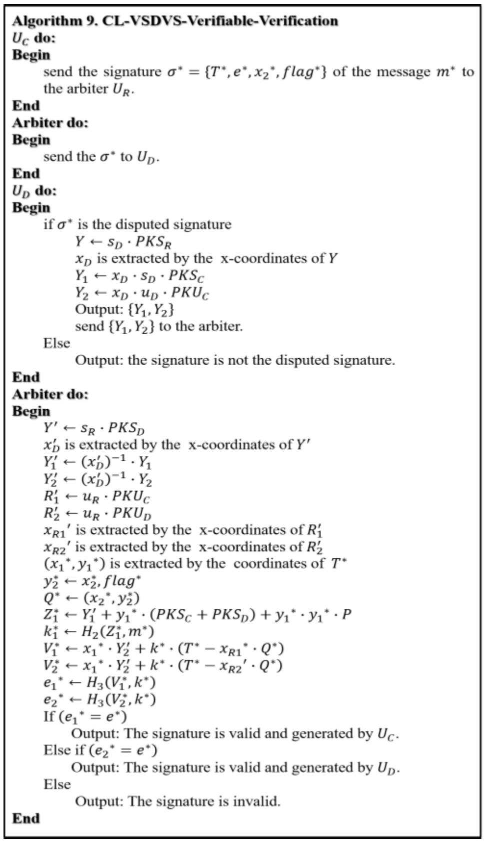
In this section, we will provide security analysis of the new scheme.

A. SECURITY PROOF

1) VERIFIABILITY

Theorem 1: The proposed certificateless verifiable strong designated verifier signature scheme is verifiable when the signer and the verifier dispute.

Proof: When the signer U_A wants to generate a signature about the designated verifier U_B, the signer will negotiate a secret value x_R using the arbiter U_R's public key.



```
Algorithm 9. CL-VSDVS-Verifiable-Verification
$U_C$ do:
Begin
    send the signature $\sigma^* = \{T^*, e^*, {x_2}^*, flag^*\}$ of the message $m^*$ to
    the arbiter $U_R$.
End
Arbiter do:
Begin
    send the $\sigma^*$ to $U_D$.
End
$U_D$ do:
Begin
    if $\sigma^*$ is the disputed signature
        $Y \leftarrow s_D \cdot PKS_R$
        $x_D$ is extracted by the x-coordinates of $Y$
        $Y_1 \leftarrow x_D \cdot s_D \cdot PKS_C$
        $Y_2 \leftarrow x_D \cdot u_D \cdot PKU_C$
        Output: $\{Y_1, Y_2\}$
        send $\{Y_1, Y_2\}$ to the arbiter.
    Else
        Output: the signature is not the disputed signature.
End
Arbiter do:
Begin
    $Y' \leftarrow s_R \cdot PKS_D$
    $x_D'$ is extracted by the x-coordinates of $Y'$
    $Y_1' \leftarrow (x_D')^{-1} \cdot Y_1$
    $Y_2' \leftarrow (x_D')^{-1} \cdot Y_2$
    $R_1' \leftarrow u_R \cdot PKU_C$
    $R_2' \leftarrow u_R \cdot PKU_D$
    ${x_{R1}}'$ is extracted by the x-coordinates of $R_1'$
    ${x_{R2}}'$ is extracted by the x-coordinates of $R_2'$
    $({x_1}^*, {y_1}^*)$ is extracted by the coordinates of $T^*$
    $y_2^* \leftarrow x_2^*, flag^*$
    $Q^* \leftarrow ({x_2}^*, y_2^*)$
    $Z_1^* \leftarrow Y_1' + {y_1}^* \cdot (PKS_C + PKS_D) + {y_1}^* \cdot {y_1}^* \cdot P$
    $k_1^* \leftarrow H_2(Z_1^*, m^*)$
    $V_1^* \leftarrow {x_1}^* \cdot Y_2' + k^* \cdot (T^* - {x_{R1}}^* \cdot Q^*)$
    $V_2^* \leftarrow {x_1}^* \cdot Y_2' + k^* \cdot (T^* - {x_{R2}}' \cdot Q^*)$
    ${e_1}^* \leftarrow H_3(V_1^*, k^*)$
    ${e_2}^* \leftarrow H_3(V_2^*, k^*)$
    If $({e_1}^* = e^*)$
        Output: The signature is valid and generated by $U_C$.
    Else if $({e_2}^* = e^*)$
        Output: The signature is valid and generated by $U_D$.
    Else
        Output: The signature is invalid.
End
```

FIGURE 12. Algorithm 9.

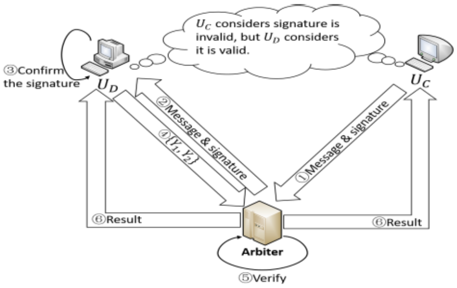


FIGURE 13. Arbitration.

And x_R will be used in the signature generation as $T = q \cdot (x_R \cdot Ps + PKU_B)$, but in the verification of the signature, according to the verification algorithm, the designated verifier U_B cannot obtain the value of x_R by T based on the hardness of ECDLP. However, the arbiter U_R can use his private key and the signer U_A's public key to get the secret value x_R. Then the arbiter U_R can verify the controversial signature by the x_R and {Y_1,Y_2 }. While when the verifier U_B wants to generate a transcript of the signature generated by U_A, the verifier will negotiate a secret value (x_R)- using the arbiter U_R's public key. And x_R will be used in the signature simulation as $\overline{T} = \overline{q} \cdot (\overline{x_R} \cdot Ps + PKU_B)$. Therefore, the arbiter can judge whether the signature generated by the signer or the verifier or others by the secret value x_R and $\overline{x_R}$.

In each case, one party will confirm whether the other party sends the corresponding signature to the arbiter to prevent the sending party from sending the wrong signature to affect the arbitration results.

When the signer insists the signature is valid but the verifier insists the signature is invalid, if the signer sending the wrong $\{Y_1, Y_2\}$ to the arbiter, it may lead to the failure of the disputed signature verification and the arbiter judges that the verifier is correct. In addition, when the signer insists the signature is invalid but the verifier insists the signature is valid, if the verifier sends the wrong $\{Y_1, Y_2\}$ to the arbiter, it may lead to the failure of the disputed signature verification and the arbiter judges that the signer is correct. Therefore, this method can effectively prevent the signer or verifier from intentionally sending the wrong $\{Y_1, Y_2\}$ to affect the arbiter's true judgment.

2) NON-TRANSFERABILITY

Theorem 2: The proposed certificateless verifiable designated verifier signature scheme is non-transferable.

Proof: Non-transferability means that it is not computationally feasible to determine who is the signer or verifier. In our scheme, signatures and transcripts can be verified, and they are indistinguishable. Therefore, an adversary can whonot judge is the signer or verifier based on the signature. Even if there is a dispute between the signer and the verifier, the arbiter needs the $\{Y_1, Y_2\}$ sent by the signer or the verifier and the secret value x_R or $\overline{x_R}$ negotiated by himself and the signer to verify the disputed signature. Besides, in the arbitration, the arbiter needs to calculate $\left\{Y_1', Y_1'\right\}$ by his private key which are protected by the difficulty of the ECDLP. Therefore, even if an adversary intercepts$\{Y_1, Y_2\}$, it is impossible to judge who is the signer or verifier. Therefore, our scheme is non-transferable.

In addition, our scheme can judge who generated the signature, but the arbiter cannot tell who is the signer or verifier.

3) NON-DELEGATABILITY

Theorem 3: The proposed certificateless verifiable strong designated verifier signature scheme is non-delegatable.

Proof: Non-delegatability means that only signers and verifiers can generate valid signatures, and no other third party can generate valid signatures. The adversary cannot know the private key of the signer or the verifier, so he cannot generate a valid signature unless he can solve the ECDL problem. If the adversary wants to forge a valid signature, first he has to know the parameters x_R which satisfies $R = u_A \cdot PKU_R$ negotiated by the signer and the arbiter. Then he needs to calculate $Z = (s_A + y_1) \cdot (PKS_B + y_1 \cdot P)$ and $V = (x_1 \cdot u_A + q \cdot k) \cdot PKU_B$. However, he can't get the signer's private key, so he can't forge a valid signature.

Even if the adversary expects to forge a transcript that is indistinguishable from the original signature, he needs to compute $\overline{V} = u_B \cdot \overline{x_1} \cdot PKU_A + u_B \cdot \overline{k} \cdot \overline{Q}$, but he cannot solve the ECDL problem to obtain the verifier's private key.

In another case, when the signer and verifier request arbitration, the adversary cannot obtain the private key of the arbiter, he cannot computes $\left\{Y_1'=s_A \cdot s_B \cdot P, Y_2'=u_A \cdot u_B \cdot Ps\right\}$ according to $\{Y_1, Y_2\}$. Thus, the adversary cannot calculate $Z_1^* = Y_1' + y_1^* \cdot (PKS_A + PKS_B) + y_1^* \cdot y_1^* \cdot P$ and $V^* = x_1^* \cdot Y_2' + k^* \cdot q \cdot PKU_B$ the private key of the arbiter.

4) UNFORGEABILITY

Theorem 4: In the random oracle model, if the difficulty of ECDLP exists, the proposed scheme is existentially unforgeable against Type I adversary under adaptive chosen-message attacks.

If there is a polynomial time super-level Type I adversary who has the non-negligible advantage ε to forge a valid CL-VSDVS, and there must be a polynomial time challenger has the advantage ε_1 which satisfies $\varepsilon_1 \geq \varepsilon \cdot \frac{\mathrm{t_{cu}}-1}{\mathrm{t_{cu}} \cdot \mathrm{t_{cu}}} \left(1-\frac{t_{cu}}{n}\right)^{t_{cu}} \left(1-\frac{1}{\mathrm{t}_{cu}}\right)^{\mathrm{t}_{ppke}} \left(1-\frac{1}{\mathrm{t}_{h2}}\right)^{t_v} \left(1-\frac{1}{t_{h3}}\right)^{t_v}$ to solve the ECDLP.

Proof: Assume that a probabilistic polynomial-time algorithm super-level adversary $\mathcal{A}$ has non-negligible advantage ε to forge a valid CL-VSDVS of the propose scheme. We define an example of a challenger $\mathcal{C}$ solving an ECDL problem as (G_1, P, sP). We set the target ID to ID_t, and the goal is for the challenger to calculate the value of s by interacting with the adversary. The interactions between the adversary $\mathcal{A}$ and the challenge $\mathcal{C}$ are described below:

-Setup: The challenger $\mathcal{C}$ inputs a security parameter l, runs the Setup algorithm, and generates the system parameters $\left\{E_p\left(a,b\right), G_1, n, P, H_1, H_2, H_3\right\}$. The challenger $\mathcal{C}$ returns the public system parameters $\left\{E_p\left(a,b\right), G_1, n, P, Ps, H_1,\right.$ $\left.H_2, H_3\right\}$ to the adversary $\mathcal{A}$ where $Ps = s \cdot P$ but s is unknown.

-Hash-Query to H_1: The adversary $\mathcal{A}$ inputs $\{ID_i, D_i\}$ to request H_1query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $h_i \in [1, n-1]$, and returns h_i to the adversary. Finally, $\mathcal{C}$ adds (ID_i, D_i, h_i) to H_1_list. If $\mathcal{C}$ finds the corresponding record in the H_1_list, then returns h_i to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Hash-Query to H_2: The adversary $\mathcal{A}$ inputs $\{Z, m\}$ to request H_2query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $k \in [1, n-1]$, and returns k to the adversary. Finally, $\mathcal{C}$ adds $\{\mathrm{Z}, m, k\}$ to H_2_list. If $\mathcal{C}$ finds the corresponding record in the H_2_list, then returns k to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Hash-Query to H_3: The adversary $\mathcal{A}$ inputs $\{V, k\}$ to request H_3query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $e \in [1, n-1]$, and returns k to the adversary. Finally, $\mathcal{C}$ adds (V, k, e) to H_3_list. If $\mathcal{C}$ finds the corresponding record in the H_3_list, then returns e to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Create-User: The adversary $\mathcal{A}$ sends a user's ID to the challenger $\mathcal{C}$, if $ID_i = ID_t$, the challenger randomly selects $r_t, h_t \in [1, n-1]$, adds (ID_t, h_t) to the H_1_list. $\mathcal{C}$ computes $D_t = r_t \cdot P$, $s_t = \perp$, adds $(ID_t, \perp, D_t)$ to the *Partial_private_key_list*. And then $\mathcal{C}$ picks the secret value $u_t \in [1, n-1]$, adds (ID_t, u_t) to the *Secret_value_list*. $\mathcal{C}$ computes $PKU_t = u_t \cdot Ps$, sets $PKS_t = D_t + h_t \cdot Ps$, and adds (ID_i, PKU_i, PKS_i) to the *Public_key_list*.

If $ID_i \neq ID_t$, the challenger randomly selects $s_i, r_i, h_i \in [1, n-1]$, adds (ID_i, h_i) to the H_1_list. $\mathcal{C}$ computes $D_i = s_i \cdot P - h_i \cdot Ps$, adds (ID_i, s_i, D_i) to the *Partial_private_key_list*. And then $\mathcal{C}$ picks the secret value $u_i \in [1, n-1]$, adds (ID_i, u_i) to the *Secret_value_list*. $\mathcal{C}$ computes $PKU_i = u_i \cdot Ps$, sets $PKS_i = s_i \cdot P$, and adds (ID_i, PKU_i, PKS_i) to the *Public_key_list*.

Finally, $\mathcal{C}$ adds $(ID_i, s_i, u_i, PKU_i, PKS_i)$ to *user_list*.

If the ID_i already exists in the *user_list*, $\mathcal{C}$ does not need to perform the above steps.

-Partial-Private-key-Query: The adversary $\mathcal{A}$ inputs ID_i to request Partial-private-key-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs the Create-Users algorithm, and returns s_i to the adversary $\mathcal{A}$. Finally, $\mathcal{C}$ adds(ID_i, s_i, D_i) to *Partial_private_key_list*.

If $ID_i = ID_t$, because of $s_t = \perp$, $\mathcal{C}$ ends the game.

If $\mathcal{C}$ finds the corresponding record in the *Partial_private_key_list*, then returns s_i to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Secret-Value-Query: The adversary $\mathcal{A}$ inputs ID_i to request Secret-value-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $u_i \in [1, n-1]$, and returns u_i to the adversary. Finally, $\mathcal{C}$ adds (ID_i, u_i) to *Secret_value_list*.

If $\mathcal{C}$ finds the corresponding record in the *Secret_value_list*, then returns u_i to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Public-key-Query: The adversary $\mathcal{A}$ inputs ID_i to request Public-key-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs the Secret-value-query, Partial-private-key-query to obtain the private key $sk_i = \{u_i, s_i\}$, and then runs Set-Public-key algorithm to obtain the public key $\{PKU_i, PKS_i\}$. Finally, $\mathcal{C}$ returns the public key $\{PKU_i, PKS_i\}$ to $\mathcal{A}$ and adds (ID_i, PKU_i, PKS_i) to *Public_key_list*.

If $\mathcal{C}$ finds the corresponding record in the *Public_key_list*, then returns $\{PKU_i, PKS_i\}$ to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Replace-Public-key-Query: The adversary $\mathcal{A}$ inputs $\left\{ID_i, PKU_i, PKS_i, PKU_i', PKS_i'\right\}$ to request Replace-public-key-query from the challenger $\mathcal{C}$. Note that the ID_i is an existing user's ID. $\mathcal{C}$ sets $u_i = \perp$ and updates the *Public_key_list* and *Secret_value_list*.

-CL-VSDVS-Generation-Query: The adversary $\mathcal{A}$ inputs $\left\{m, ID_i, ID_j\right\}$ to request CL-VSDVS-Generation-query from the challenger $\mathcal{C}$. $\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_i, and runs the Public-key-query to get the public key of the ID_j. Then $\mathcal{C}$ randomly selects $q \in [1, n-1]$ and executes the CL-VSDVS-Generation algorithm to obtain the corresponding signature σ.

If $ID_i = ID_t$ or ID_i has been submitted to a public key replacement query by the adversary, in other words, $u_i = \perp$ or $s_i = \perp$.$\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_j and randomly selects $T \in G_1, q, k, x_R \in Z_n^*$, then computes $Q = q \cdot Ps$ and

$Z = \left(s_j + y_1\right) \cdot (PKS_t + y_1 \cdot P)$ where (x_1, y_1) is extracted by the coordinates of T. Then $\mathcal{C}$ asks $k = H_2(Z, m)$ query and computes $V = x_1 \cdot u_j \cdot PKU_t + q \cdot k \cdot u_j$. Finally, $\mathcal{C}$ asks for $e = H_3(V, k)$ and returns $\sigma = \{T, e, x_2, flag\}$ where (x_2, y_2) is extracted by the coordinates of Q and *flag* depends on the parity of y_2. Apparently, $\sigma = \{T, e, x_2, flag\}$ can satisfy the verification.

-CL-VSDVS-Verification-Query: The adversary $\mathcal{A}$ inputs $\left\{m, \sigma, ID_i, ID_j\right\}$ to request CL-VSDVS-Verification query from the challenger $\mathcal{C}$.

If $ID_i \neq ID_t$ (one case is $ID_j = ID_t$), $\mathcal{C}$ ends the game.

Otherwise, $\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_j, and runs the Public-key-query to get the public key of the ID_i. Then $\mathcal{C}$ executes the CL-VSDVS-Verification algorithm to verify the validity of the signature σ. If the signature σ can be verified, this means that $\mathcal{C}$ can find $Z = \left(s_j + y_1\right) \cdot (PKS_t + y_1 \cdot P)$ in the H_2_list, $V = x_1 \cdot u_j \cdot PKU_t + q \cdot k \cdot u_j$, k, and e in the H_3_list. If $\mathcal{C}$ cannot search for any of the four, stop the game.

Forgery: The adversary $\mathcal{A}$ gives a quaternion $\left(m', \sigma', ID'_i, ID'_j\right)$ which satisfy:

① The signature σ^* is the valid signature of ID^*_i, ID^*_j and the message m^*. The adversary $\mathcal{A}$ has never submitted the CL-VSDVS-Generation-query of ID^*_i, ID^*_j and m^* to the challenger $\mathcal{C}$.

② The adversary $\mathcal{A}$ has never submitted Partial-private-key-query and Secret-value-query about ID^*_i, ID^*_j to challenger $\mathcal{C}$.

If $ID'_i \neq ID_t$ (one case is $ID'_j = ID_t$), $\mathcal{C}$ ends the game. Otherwise, $\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_j, and runs the Public-key-query to get the public key of the ID_i. Then $\mathcal{C}$ executes the CL-VSDVS-Verification algorithm to verify the valid of the signature σ'. If the signature σ can be verified, this means that $\mathcal{C}$ can find $Z = \left(s_j + y'_1\right) \cdot \left(PKS_t + y'_1 \cdot P\right)$ in the H_2_list and $V = x_1 \cdot u_j \cdot PKU_t + q \cdot k \cdot u_j, k, e$ in the H_3_list. If $\mathcal{C}$ cannot search for any of the four, stop the game. Conversely, if σ' is valid, set $t'_1 \cdot P = \left(s_j + y'_1\right) \cdot \left(PKS_t + y'_1 \cdot P\right)$ and $t'_2 \cdot P = x_1 \cdot u_j \cdot PKU_t + q \cdot k \cdot u_j$. From the game, we can get that $s_t = r_t + h_t \cdot s$ and $PKU'_j = u'_j \cdot Ps = u'_j \cdot s \cdot P$, so we can know that $t'_1 = \left(s_j + y'_1\right) \cdot \left(r_t + h_t \cdot s + y'_1\right)$ and $t'_2 = u_j \cdot x'_1 \cdot u_t \cdot s + u_j \cdot k' \cdot q \cdot s$.

According to the Forking Lemma [58], $\mathcal{C}$ can also use $t'_1 = \left(s_j + y_1\right) \cdot h_t \cdot s + \left(s_j + y_1\right) \cdot (r_t + y_1)$ to replay the game with the same random tape but different outputs of $H_1 query$, and $\mathcal{C}$ can obtain the second equation:

$$t'_1 = \left(s_j + y_1\right) \cdot h_t^{(2)} \cdot s + \left(s_j + y_1\right) \cdot (r_t + y_1),$$

Because $\mathcal{C}$ can know x'_1 from Q', s_j andr_t, but cannot know t'_1. $\mathcal{C}$ can get the value of s and t'_1 according to the two equations. In addition, $\mathcal{C}$ can also use $t'_2 = x'_1 \cdot u_t \cdot u'_j \cdot s + q' \cdot k' \cdot u'_j \cdot s$ to replay the game with same random tape but different outputs of $H_2 query$, and $\mathcal{C}$ can obtain the second equation:

$$t'_2 = x'_1 \cdot u_t \cdot u'_j \cdot s + q' \cdot k^{(2)} \cdot u'_j \cdot s,$$

Because $\mathcal{C}$ can know x'_1 from Q', q', u_t and u'_j, $\mathcal{C}$ can get the value of s and t'_2 according to the two equations.

Analysis: In order to calculate the probability of $\mathcal{C}$ winning the game, we define the following events:

E1: In Create-User algorithms, the identity ID_t can be created.

E2: In the Partial-private-key-query,$\mathcal{C}$ did not terminate the game.

E3: Z in the H_2_list

E4: V, k, e in the H_3_list

E5: $\mathcal{A}$ forges a valid signature σ^* of the identities ID_t, ID^*_j and message m^*.

According to the simulation of the game, we can assume t_{cu}, t_{ppke}, $t_v \mathrm{t}_{h1}$, t_{h2}, t_{h3} are the maximum times of the Create-Users, the Partial-private-key-query, the CL-VSDVS-Verification-query, the Hash-query to H_1, the Hash-query to H_2 and the Hash-query to H_3, respectively. n is the number of elements in the G_1. And we can derive

$$\Pr[E1] \geq \left(1 - \frac{t_{cu}}{n}\right)^{t_{cu}},$$

$$\Pr[E2|E1] \geq \left(1 - \frac{1}{\mathrm{t}_{cu}}\right)^{\mathrm{t}_{ppke}},$$

$$\Pr\left[E3|E2 \bigwedge E1\right] \geq \left(1 - \frac{1}{\mathrm{t}_{h2}}\right)^{t_v},$$

$$\Pr\left[E4|E3 \bigwedge E2 \bigwedge E1\right] \geq \left(1 - \frac{1}{t_{h3}}\right)^{t_v},$$

$$\Pr\left[E5|E4 \bigwedge E3 \bigwedge E2 \bigwedge E1\right] \geq \varepsilon \cdot \frac{1}{\mathrm{t}_{cu}} \cdot \frac{\mathrm{t}_{cu} - 1}{\mathrm{t}_{cu}},$$

The probability that $\mathcal{C}$ can win the game is

$$\Pr\left[E5 \bigwedge E4 \bigwedge E3 \bigwedge E2 \bigwedge E1\right] \geq \varepsilon \cdot \frac{\mathrm{t}_{cu} - 1}{\mathrm{t}_{cu} \cdot \mathrm{t}_{cu}} \left(1 - \frac{t_{cu}}{n}\right)^{t_{cu}} \left(1 - \frac{1}{\mathrm{t}_{cu}}\right)^{\mathrm{t}_{ppke}} \left(1 - \frac{1}{\mathrm{t}_{h2}}\right)^{t_v} \times \left(1 - \frac{1}{t_{h3}}\right)^{t_v}.$$

Because of the difficulty of ECDLP, the proposed scheme is existentially unforgeable against Type I adversary under adaptive chosen-message attacks.

Theorem 5: In the random oracle model, if the difficulty of ECDLP exists, the proposed scheme is existentially unforgeable against Type II adversary under adaptive chosen-message attacks.

If there is a polynomial time super Type II adversary who has the non-negligible advantage ε to forge a valid CL-VSDVS, and there must be a polynomial time challenger has the advantage ε_1 which satisfies

$$\varepsilon_1 \geq \varepsilon \cdot \frac{\mathrm{t}_{cu} - 1}{\mathrm{t}_{cu} \cdot \mathrm{t}_{cu}} \left(1 - \frac{t_{cu}}{n}\right)^{t_{cu}} \left(1 - \frac{1}{\mathrm{t}_{cu}}\right)^{\mathrm{t}_{svq}} \left(1 - \frac{1}{\mathrm{t}_{h2}}\right)^{t_v} \times \left(1 - \frac{1}{t_{h3}}\right)^{t_v}$$

to solve the ECDLP.

Proof: Assume that a probabilistic polynomial-time algorithm super-level type II adversary $\mathcal{A}$ has non-negligible advantage ε to forge a valid CL-VSDVS of the proposed scheme. We define an example of a challenger $\mathcal{C}$ solving an ECDL problem as (G_1, P, uP). We set the target ID to ID_t, and the goal is for the challenger to calculate the value of u by interacting with the adversary. The interactions between the adversary $\mathcal{A}$ and the challenge $\mathcal{C}$ are described below:

-Setup: The challenger $\mathcal{C}$ inputs a security parameter l, runs the Setup algorithm, and generates the master-private key s, master-public key $Ps = s \cdot P$ and other system parameters $\left\{E_p\left(a, b\right), G_1, n, P, H_1, H_2, H_3\right\}$. The challenger $\mathcal{C}$ returns the master-private key s and the public system parameters $\left\{E_p\left(a, b\right), G_1, n, P, Ps, H_1, H_2, H_3\right\}$ to the adversary $\mathcal{A}$.

-Hash-Query to H_1: The adversary $\mathcal{A}$ inputs $\{ID_i, D_i\}$ to request $H_1 query$ from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $h_i \in [1, n-1]$, and returns k to the adversary. Finally, $\mathcal{C}$ adds (ID_i, D_i, h_i) to H_1_list. If $\mathcal{C}$ finds the corresponding record in the H_1_list, then returns h_i to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Hash-Query to H_2: The adversary $\mathcal{A}$ inputs $\{\mathrm{V}, m\}$ to request $H_2 query$ from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $k \in [1, n-1]$, and returns k to the adversary. Finally, $\mathcal{C}$ adds $\{V, m, k\}$ to H_2_list. If $\mathcal{C}$ finds the corresponding record in the H_2_list, then returns k to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Hash-Query to H_3: The adversary $\mathcal{A}$ inputs $\{V, k\}$ to request $H_3 query$ from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $e \in [1, n-1]$, and returns k to the adversary. Finally, $\mathcal{C}$ adds $\{V, k, e\}$ to H_3_list. If $\mathcal{C}$ finds the corresponding record in the H_3_list, then returns e to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Create-User: The adversary $\mathcal{A}$ sends a user's ID to the challenger $\mathcal{C}$, the challenger randomly selects $s_i, r_i, h_i \in [1, n-1]$, adds (ID_i, h_i) to the H_1_list. $\mathcal{C}$ computes $D_i = r_i \cdot P$, $s_i = r_i + h_t \cdot s$, then adds (ID_i, s_i, D_i) to the $Partial\_private\_key\_list$.

If $ID_i = ID_t$, the challenger sets $u_t = \perp$, adds (ID_t, u_t) to the $Secret\_value\_list$. $\mathcal{C}$ computes $PKU_t = uP \cdot \mathrm{s}$, $PKS_t = s_t \cdot P$, and adds (ID_t, PKU_t, PKS_t) to the $Public\_key\_list$.

If $ID_i \neq ID_t$, And then $\mathcal{C}$ picks the secret value $u_i \in [1, n-1]$, adds (ID_i, u_i) to the $Secret\_value\_list$. $\mathcal{C}$ computes $PKU_i = u_i \cdot Ps$, sets $PKS_i = s_i \cdot P$, and adds (ID_i, PKU_i, PKS_i) to the $Public\_key\_list$.

Finally, $\mathcal{C}$ adds $(ID_i, s_i, u_i, PKU_i, PKS_i)$ to $user\_list$

If the ID_i already exists in the $user\_list$, $\mathcal{C}$ does not need to perform the above steps.

-Partial-Private-key-Query: The adversary $\mathcal{A}$ inputs ID_i to request Partial-private-key-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs the Create-Users algorithm, and returns s_i to the adversary. Finally, $\mathcal{C}$ adds(ID_i, s_i, D_i) to $Partial\_private\_key\_list$. If $\mathcal{C}$ finds the corresponding record in the $Partial\_private\_key\_list$, then returns s_i to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Secret-Value-Query: The adversary $\mathcal{A}$ inputs ID_i to request Secret-value-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ randomly selects $u_i \in [1, n-1]$, and returns u_i to the adversary. Finally, $\mathcal{C}$ adds (ID_i, u_i) to $Secret\_value\_list$.

If $ID_i = ID_t$, because of $u_t = \perp$, $\mathcal{C}$ ends the game.

If $\mathcal{C}$ finds the corresponding record in the $Secret\_value\_list$, then returns u_i to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-Public-key-Query: The adversary $\mathcal{A}$ inputs ID_i to request Public-key-query from the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs the Secret-value-query, Partial-private-key-query to obtain the private key $sk_i = \{u_i, s_i\}$, and then runs Set-Public-key algorithm to obtain the public key $\{PKU_i, PKS_i\}$. Finally, $\mathcal{C}$ returns the public key $\{PKU_i, PKS_i\}$ to $\mathcal{A}$ and adds (ID_i, PKU_i, PKS_i) to $Public\_key\_list$.

If $ID_i = ID_t$, $\mathcal{C}$ returns $\{PKU_t = uP \cdot s, PKS_t = s_t \cdot P\}$ directly.

If $\mathcal{C}$ finds the corresponding record in the $Public\_key\_list$, then returns $\{PKU_i, PKS_i\}$ to $\mathcal{A}$, $\mathcal{C}$ does not need to perform the above steps.

-CL-VSDVS-Generation-Query: The adversary $\mathcal{A}$ inputs $\left\{m, ID_i, ID_j\right\}$ to request CL-VSDVS-Generation-query from the challenger $\mathcal{C}$. $\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_i, and runs the Public-key-query to get the public key of the ID_j. Then $\mathcal{C}$ randomly selects $q \in [1, n-1]$ and executes the CL-VSDVS-Generation algorithm to obtain the corresponding signature σ.

If $ID_i = ID_t$ or ID_i has been submitted to a public key replacement query by the adversary, in other words, $u_i = \perp$ or $s_i = \perp$.$\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_j and randomly selects $T \in G_1, q, k, x_R \in Z_n^*$, then computes $Q = q \cdot Ps$ and $Z = \left(s_j + y_1\right) \cdot (PKS_t + y_1 \cdot P)$ where (x_1, y_1) is extracted by the coordinates of T. Then $\mathcal{C}$ asks for $k = H_2\left(Z, m\right)$ query and computes $V = x_1 \cdot u_j \cdot PKU_t + q \cdot k \cdot u_j$. Finally, $\mathcal{C}$ asks for $e = H_3\left(V, k\right)$ and returns $\sigma = \{T, e, x_2, flag\}$ where (x_2, y_2) is extracted by the coordinates of Q and $flag$ depends on the parity of y_2. Apparently, $\sigma = \{T, e, x_2, flag\}$ can satisfy the verification.

-CL-VSDVS-Verification-Query: The adversary $\mathcal{A}$ inputs $\left\{m, \sigma, ID_i, ID_j\right\}$ to request CL-VSDVS-Verification-query from the challenger $\mathcal{C}$.

If $ID_i \neq ID_t$ (one case is $ID_j = ID_t$), $\mathcal{C}$ ends the game.

Otherwise, $\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_j, and runs the Public-key-query to get the public key of the ID_i. Then $\mathcal{C}$ executes the CL-VSDVS-Verification algorithm to verify the validity of the signature σ. If the signature σ can be verified, this means that $\mathcal{C}$ can find $Z = \left(s_j + y_1\right) \cdot (PKS_t + y_1 \cdot P)$ in the H_2_list, $V = x_1 \cdot u_j \cdot PKU_t + q \cdot k \cdot u_j$, k, and e in the H_3_list. If $\mathcal{C}$ cannot search for any of the four, stop the game.

Forgery: Finally, the adversary $\mathcal{A}$ gives a quaternion $\left(m', \sigma', ID_i', ID_j'\right)$ which satisfies:

① The signature σ^* is the valid signature of ID_i^*, ID_j^* and the message m^*. The adversary $\mathcal{A}$ has never submitted the CL-VSDVS-Generation-query of ID_i^*, ID_j^* and m^* to the challenger $\mathcal{C}$.

② The adversary $\mathcal{A}$ has never submitted Secret-value-query about ID_i^*, ID_j^* to challenger $\mathcal{C}$.

If $ID_i' \neq ID_t$ (one case is $ID_j' = ID_t$), $\mathcal{C}$ ends the game. Otherwise, $\mathcal{C}$ runs the Partial-private-key-query and Secret-value-query to obtain the private key of the ID_j, and runs the Public-key-query to get the public key of the ID_i. Then $\mathcal{C}$ executes the CL-VSDVS-Verification algorithm to verify the valid of the signature σ'. If the signature σ can be verified, this means that $\mathcal{C}$ can find $Z = \left(s_j + y_1'\right) \cdot \left(PKS_t + y_1' \cdot P\right)$ in the H_2_list and $V = x_1 \cdot u_j{\cdot}PKU_t + q \cdot k \cdot u_j, k, e$ in the H_3_list. If $\mathcal{C}$ cannot search for any of the four, stop the game. Conversely, if σ' is valid, set and $t_2' \cdot P = x_1 \cdot u_j{\cdot}PKU_t + q \cdot k \cdot u_j$. From the game, we can get that $PKU_j' = u_j' \cdot Ps = u_j' \cdot s \cdot P$, so we can know that $t_2' = u_j \cdot x_1' \cdot u_t \cdot s + u_j \cdot k' \cdot q \cdot s$. According to the Forking Lemma [58], $\mathcal{C}$ can use $t_2' = x_1' \cdot u_t \cdot u_j' \cdot s + q' \cdot k' \cdot u_j' \cdot s$ to replay the game with the same random tape but different outputs of H_2query, and $\mathcal{C}$ can obtain the following equation:

$$t_2' = x_1' \cdot u_t \cdot u_j' \cdot s + q' \cdot k^{(2)} \cdot u_j' \cdot s,$$

Because $\mathcal{C}$ can know x_1' from Q', s, q', u_t and u_j', but cannot know t_2'. $\mathcal{C}$ can get the value of s and t_2' according to the two equations.

Analysis: In order to calculate the probability of $\mathcal{C}$ winning the game, we define the following events:

E1: In Create-User algorithms, the identity ID_t can be created.
E2: In the Secret-value-query, $\mathcal{C}$ did not terminate the game.
E3: Z in the H_2_list.
E4: V, k, e in the H_3_list.
E5: $\mathcal{A}$ forgery a valid signature σ^* the identities ID_t, ID_j^* and message m^*.

According to the simulation of the game, we can assume t_{cu}, t_{svq}, $t_v\mathrm{t}_{h1}$, t_{h2}, t_{h3} are the maximum times of the Create-Users, the Secret-value-query, the CL-VSDVS-Verification-query, the Hash-query to H_1, the Hash-query to H_2 and the Hash-query to H_3, respectively. n is the number of elements in the G_1. And we can derive

$$\Pr[E1] \geq \left(1 - \frac{t_{cu}}{n}\right)^{t_{cu}},$$

$$\Pr[E2|E1] \geq \left(1 - \frac{1}{\mathrm{t}_{cu}}\right)^{\mathrm{t}_{svq}},$$

$$\Pr\left[E3|E2 \bigwedge E1\right] \geq \left(1 - \frac{1}{\mathrm{t}_{h2}}\right)^{t_v},$$

$$\Pr\left[E4|E3 \bigwedge E2 \bigwedge E1\right] \geq \left(1 - \frac{1}{t_{h3}}\right)^{t_v},$$

$$\Pr\left[E5|E4 \bigwedge E3 \bigwedge E2 \bigwedge E1\right] \geq \varepsilon \cdot \frac{1}{\mathrm{t_{cu}}} \cdot \frac{\mathrm{t_{cu}} - 1}{\mathrm{t_{cu}}}.$$

The probability that $\mathcal{C}$ can win the game is

$$\Pr\left[E5 \bigwedge E4 \bigwedge E3 \bigwedge E2 \bigwedge E1\right]$$
$$\geq \varepsilon_1 \geq \varepsilon \cdot \frac{\mathrm{t_{cu}} - 1}{\mathrm{t_{cu}} \cdot \mathrm{t_{cu}}} \left(1 - \frac{t_{cu}}{n}\right)^{t_{cu}} \left(1 - \frac{1}{\mathrm{t}_{cu}}\right)^{\mathrm{t}_{svq}} \left(1 - \frac{1}{\mathrm{t}_{h2}}\right)^{t_v}$$
$$\times \left(1 - \frac{1}{t_{h3}}\right)^{t_v}.$$

Because of the difficulty of ECDLP, the proposed scheme is existentially unforgeable against Type II adversary under adaptive chosen-message attacks.

5) SIGNER AMBIGUITY

Assuming that $\sigma^* = \left\{T^*, e^*, x_2^*, flag^*\right\}$ is a valid signature for the message m of the signer U_A and verifier U_B. Because a random number $q \in [1, n-1]$ is needed in the signature generation, the difference of q will lead to the difference of valid signatures. Thus, the signer generates a valid signature $\sigma = \{T, e, x_2, flag\}$ for the message m, and we can know that $\Pr\left[\sigma = \sigma^*\right] = \frac{1}{n-1}$ where n is the number of elements in the G_1. As for the transcripts, in CL-VSDVS Simulation, the verifier needs to select a random number $\overline{q} \in [1, n-1]$, the difference of $\overline{q}$ will lead to the difference of valid signatures. Thus, the verifier generates a transcript $\overline{\sigma} = \left\{\overline{T}, \overline{e}, \overline{x_2}, \overline{flag}\right\}$ for the message, and we can know that $\Pr\left[\overline{\sigma} = \sigma^*\right] = \frac{1}{n-1}$. Because the signature and the transcript are indistinguishable, the adversary cannot judge who is the signer, so our scheme can achieve signer ambiguity.

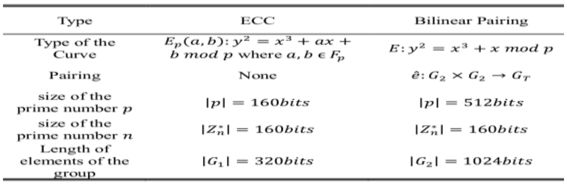
Even if the signer is attacked by the key compromise attack, the adversary can get the private key of the signer, but the adversary cannot get $q \in [1, n-1]$ which is discarded after signature generation, so the adversary cannot calculate $V = (x_1 \cdot u_A + q \cdot k) \cdot PKU_B$, and $e = H_3(V, k)$. Thus, we can know that the adversary cannot verify the signature. Because an adversary cannot obtain the verifier's private key based on the difficulty of the ECDLP and cannot verify transcript. In short, our scheme can also achieve signer ambiguity under signer's key compromise attack if there is no efficient probabilistic polynomial-time algorithm can break ECDLP.

VII. COMPARISON

In order to achieve security level of 80 bits, we will set the parameters as shown in the following Table 2. We define a non-singular elliptic curve $E_p(a, b) : y^2 = x^3 + ax + b \bmod p$ where $a, b \epsilon F_p$ and p is a 160bits prime number. G_1 is an additive cyclic group which generated by a point P on $E_p(a, b)$, its order is 160bits prime number n. We define the Tate pairing over the super singular elliptic curve $E : y^2 = x^3 + x \bmod p$ with embedding degree 2. We set n is a 160 bits Solinas prime $n = 2^{159} + 2^{17} + 1$ and a 512 bits prime number $p+1 = 12nr$.

The cost time of the operations are performed in Windows 10 with Intel(R) Core (TM) i5-4210H CPU 2.90GHz and 4.0GB RAM.

TABLE 2. Parameters used for security level of 80 bits.



Type	ECC	Bilinear Pairing
Type of the Curve	$E_p(a,b): y^2 = x^3 + ax + b \ mod \ p$ where $a, b \in F_p$	$E: y^2 = x^3 + x \ mod \ p$
Pairing	None	$\hat{e}: G_2 \times G_2 \to G_T$
size of the prime number p	$\lvert p\rvert = 160bits$	$\lvert p\rvert = 512bits$
size of the prime number n	$\lvert Z_n^*\rvert = 160bits$	$\lvert Z_n^*\rvert = 160bits$
Length of elements of the group	$\lvert G_1\rvert = 320bits$	$\lvert G_2\rvert = 1024bits$

Then we give the following definitions:

T_1 : Time to perform a hash function operation, and here we assume that the execution time of all types of hash functions is approximately $1.4ms$.

T_2 : Time to perform a pairing-based exponentiation operation. T_2 is approximately $37ms$.

T_3 : Time to perform an ECC-based scalar point multiplication operation. T_3 is approximately $23ms$.

T_4 : Time to perform a bilinear pairing operation. T_4 is approximately $53ms$.

T_5 : The time to perform an operation of calculating y on elliptic curve based on x. T_5 is approximately $8.2ms$

$|G_1|$: The length of element in G_1.

$|G_2|$: The length of elements in G_2 satisfying $G_2 \times G_2 \to G_T$.

$\left|Z_n^*\right|$: The length of element in Z_n^*.

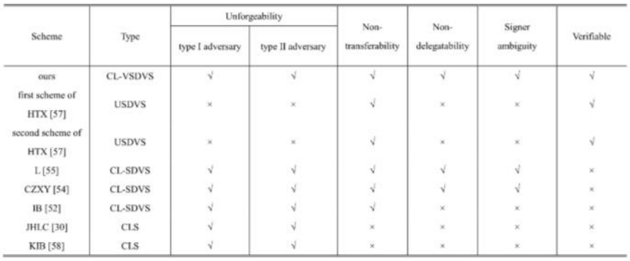
$\left|Z_m^*\right|$: The length of element in Z_m^* which satisfies Z_n^* is a subgroup of Z_m^* and $m = 2n + 1$.

A. COMPARISONS OF SECURITY

According to Table 3, we can get that our scheme and L's [54], CZXY's [53] and IB's [51] scheme can meet the security requirements in the following while the JHLC's [29] and KIB's [57] scheme are not strong designated signature schemes, so their schemes do not satisfy requirements of non-delegability, non-transferability and signer ambiguity security. In addition, both scheme of HTX [56] are not certificateless signature, thus, they cannot achieve the unforgeability against type I adversary and type II adversary. Besides, they cannot achieve non-delegatability and signer ambiguity.

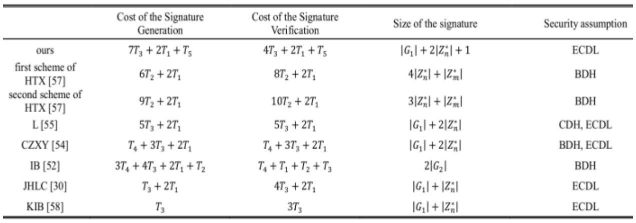
Note that for verifiability, both schemes of HTX are undeniable so that they can also achieve verifiability. However, our verifiability is different from theirs. This is because: (1) the first scheme of HTX realizes that the signer cannot deny the signature generated by himself; the second realizes both that the signer cannot deny the signature generated by himself, and that the designated verifier cannot deny the transcript generated by herself. (2) However, in our scheme, the arbiter can judge the validity of the signature, and then we can get who generated the original signature, i.e. the signer's undeniability as well as who generated the signature transcript, i.e. the verifier's unde- niability. More importantly, ours can confirm whether the verifier denies the valid signature.

TABLE 3. Security requirements of our scheme and related schemes.



Scheme	Type	Unforgeability		Non-transferability	Non-delegatability	Signer ambiguity	Verifiable
		type I adversary	type II adversary				
ours	CL-VSDVS	√	√	√	√	√	√
first scheme of HTX [57]	USDVS	×	×	√	×	×	√
second scheme of HTX [57]	USDVS	×	×	√	×	×	√
L [55]	CL-SDVS	√	√	√	√	√	×
CZXY [54]	CL-SDVS	√	√	√	√	√	×
IB [52]	CL-SDVS	√	√	√	×	×	×
JHLC [30]	CLS	√	√	×	×	×	×
KIB [58]	CLS	√	√	×	×	×	×

TABLE 4. Time complexity of our scheme and related schemes.



	Cost of the Signature Generation	Cost of the Signature Verification	Size of the signature	Security assumption
ours	$7T_3 + 2T_1 + T_5$	$4T_3 + 2T_1 + T_5$	$\lvert G_1\rvert + 2\lvert Z_n^*\rvert + 1$	ECDL
first scheme of HTX [57]	$6T_2 + 2T_1$	$8T_2 + 2T_1$	$4\lvert Z_n^*\rvert + \lvert Z_m^*\rvert$	BDH
second scheme of HTX [57]	$9T_2 + 2T_1$	$10T_2 + 2T_1$	$3\lvert Z_n^*\rvert + \lvert Z_m^*\rvert$	BDH
L [55]	$5T_3 + 2T_1$	$5T_3 + 2T_1$	$\lvert G_1\rvert + 2\lvert Z_n^*\rvert$	CDH, ECDL
CZXY [54]	$T_4 + 3T_3 + 2T_1$	$T_4 + 3T_3 + 2T_1$	$\lvert G_1\rvert + 2\lvert Z_n^*\rvert$	BDH, ECDL
IB [52]	$3T_4 + 4T_3 + 2T_1 + T_2$	$T_4 + T_1 + T_2 + T_3$	$2\lvert G_2\rvert$	BDH
JHLC [30]	$T_3 + 2T_1$	$4T_3 + 2T_1$	$\lvert G_1\rvert + \lvert Z_n^*\rvert$	ECDL
KIB [58]	T_3	$3T_3$	$\lvert G_1\rvert + \lvert Z_n^*\rvert$	ECDL

In conclusion, our scheme satisfies the requirements of very- fiability, unforgeability, non-delegability, non-transferability and signer ambiguity.

B. COMPUTATIONAL OVERHEADS

According to Table 4, we can obtain the comparisons between our scheme and other schemes in terms of time complexity. Figure 15 shows that our signature generation time of the scheme is shorter than IB's and both two schemes of the HTX, but longer than that of L's, CZXY's, JHIC's and KIB's schemes. However, compared with L's and CZXY's schemes, ours could achieve verifiability; compared with JHLC's and KIB's oness, ours has non-transferability, non-delegatability, signer ambiguity and verifiability.

In addition, according to Figure 15, the signature verification time of our scheme is shorter than L's, CZXY's, IB's and both two schemes of the HTX, but slightly longer than that of JHLC's and KIB's scheme, but our scheme is verifiable when the signer and the verifier request the arbiter for judging the validity of the signature, and whether there is cheating between signer and verifier.

C. COMMUNICATION OVERHEADS

According to the Table 4, we can obtain the comparisons between our scheme and other schemes in terms of signature length. The signature length of our scheme is shorter than IB's and both two schemes of HTX, but slightly longer than that of L scheme, CZXY scheme, IB scheme, JHLC scheme and KIB scheme, which are shown in Figure 14. But compared with L scheme and CZXY scheme, our scheme is verifiable when the signer and the verifier request the arbiter for judging the validity of the signature and whether there is cheating between signer and verifier. Compared with IB's scheme, our scheme has non-transferability, non-delegata- bility, signer

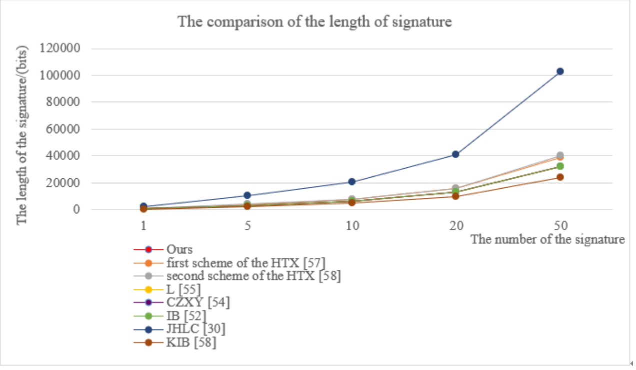
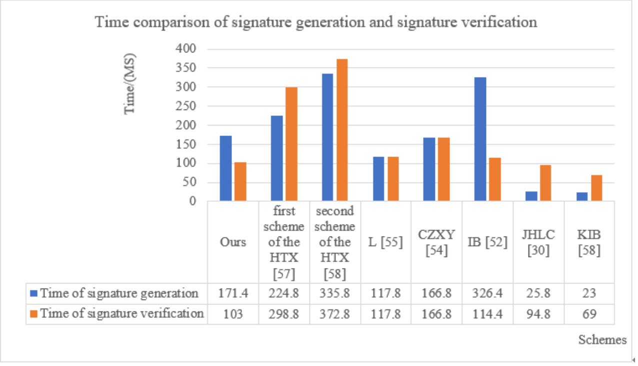


FIGURE 14. Signature length comparison.



	Ours	first scheme of the HTX [57]	second scheme of the HTX [58]	L [55]	CZXY [54]	IB [52]	JHLC [30]	KIB [58]
■ Time of signature generation	171.4	224.8	335.8	117.8	166.8	326.4	25.8	23
■ Time of signature verification	103	298.8	372.8	117.8	166.8	114.4	94.8	69

FIGURE 15. Generating signature and verifying signature time-consuming.

ambiguity and verifiability. Compared with JHLC's scheme and KIB's scheme, our scheme achieves non-transferability, non-delegatability, signer ambiguity and verifiability.

D. TRADEOFF BETWEEN VERIFIABILITY AND COMPUTATIONAL COSTS

To illustrate the tradeoffs between verifiability and computational costs, we tested at the RSA security level of 1024 bits, 2048 bits, 3072 bits, 7680bits and 15630bits, respectively, and obtained the average computational costs of multiple experiments, respectively, and obtain the average computational costs. In our scheme, the steps to achieve verifiability are in the signature generation instead of the signature verification. Therefore, we compare the time to generate signatures with verifiability, the time a signature without verifiability, and the time of CL-VSDVS-Verifiable verification to show that our solution has relatively low computational costs and being verifiable. According to our scheme, the time of generating a signature with verifiability is $7T_3 + 2T_1 + T_5$. The time of generating a signature without verifiability is $6T_3 + 2T_1 + T_5$. The time of CL-VSDVS-Verifiable verification is $16T_3 + 3T_1 + T_5$.

As shown in Figure 16, we remove the steps of achieving verifiability, and select random numbers to replace some of the parameters in the removed steps. The time of signatures generated by the five security levels without verifiability are 149ms, 203.4ms, 278.7ms, 488.7ms and 918.7ms, respectively. And the verifiable signature generation time are 172ms, 235ms and 322ms, 567.9ms and 1060.1ms, respectively. In addition, the time consumption of CL-VSDVS-Verifiable verification is 311.4ms, 521.1ms,714ms, 1255.1ms and 2352.4ms respectively. CL-VSDVS-Verifiable verification approximately takes about 2.2 times as long as the generation of verifiable signatures. Comparatively, in our scheme, although the computational cost of CL-VSDVS-Verifiable verification is higher than that of signature generation without verifiability, the gap of the signature generation time with or without verifiability is very small. Therefore, from the perspective of signature generation, our solution has relatively low computational costs with verifiability.

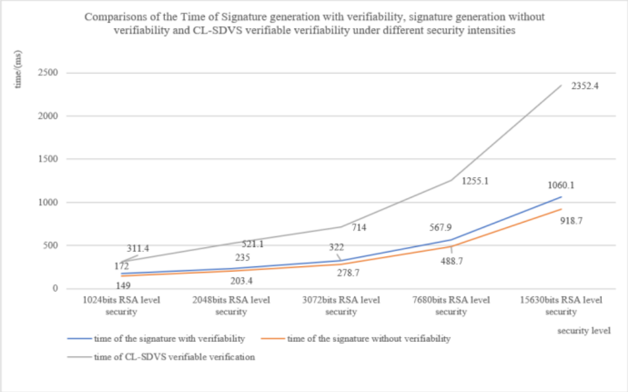


FIGURE 16. Tradeoff between verifiability and consumption.

VIII. CONCLUSION

In this paper, we proposed a new efficient certificateless strong designated verifier signature scheme without using bilinear pairings over elliptic curves. We have analyzed that the proposed scheme is non-transferable, non-delegable. In addition, we have also proved that our scheme is existentially unforgeable against type I and type II adversary under adaptive chosen-message attacks. Besides, we have proved that our scheme is unforgettable against type I and type II adversaries under adaptive chosen-message attacks. In addition, when the signer and the verifier have disputes, the scheme can effectively realize that the signer cannot deny the signature generated by himself, and the verifier cannot deny the signature verification results. In other words, the signer and the verifier can request the arbiter for judging the validity of the signature, who generated the signature and whether there is cheating between signer and verifier. Moreover, our scheme is safe and feasible for some applications in low bandwidth network communication environment. Future work will focus on using zero knowledge techniques to design certificateless verifiable strong designated verifier signature schemes.

REFERENCES

[1] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. CRYPTO*, in Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, 1985, pp. 47–53.

[2] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. ASIACRYPT*, in Lecture Notes in Computer Science, vol. 2894, Taipei, Taiwan: Springer, 2003, pp. 452–473.

[3] Y.-C. Chen and R. Tso, "A survey on security of certificateless signature schemes," *IETE Tech. Rev.*, vol. 33, no. 2, pp. 115–121, Jul. 2015.

[4] X. Huang, W. Susilo, Y. Mu, and F. Zhang, "On the security of certificateless signature schemes from Asiacrypt 2003," in *Proc. CANS*, in Lecture Notes in Computer Science, vol. 3810, Xiamen, China: Springer, 2005, pp. 13–25.

[5] D. H. Yum and P. J. Lee, "Generic construction of certificateless signature," in *Proc. ACISP*, in Lecture Notes in Computer Science, vol. 3108, Sydney, NSW, Australia: Springer, 2004, pp. 200–211.

[6] B. C. Hu, D. S. Wong, Z. Zhang, and X. Deng, "Key replacement attack against a generic construction of certificateless signature," in *Proc. ACISP*, in Lecture Notes in Computer Science, vol. 4058, Melbourne, VIC, Australia: Springer, 2006, pp. 235–246.

[7] B. C. Hu, D. S. Wong, Z. Zhang, and X. Deng, "Certificateless signature: A new security model and an improved generic construction," *Des., Codes Cryptogr.*, vol. 42, no. 2, pp. 109–126, 2007.

[8] K. Shim, "Breaking the short certificateless signature scheme," *Inf. Sci.*, vol. 179, no. 3, pp. 303–306, 2009.

[9] X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu, "Certificateless signature revisited," in *Proc. ACISP*, in Lecture Notes in Computer Science, vol. 4586, Townsville, QLD, Australia: Springer, 2007, pp. 308–322.

[10] X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu, "Certificateless signatures: New schemes and security models," *Comput. J.*, vol. 55, no. 4, pp. 457–474, Apr. 2012.

[11] R. Tso, X. Huang, and W. Susilo, "Strongly secure certificateless short signatures," *J. Syst. Softw.*, vol. 85, no. 6, pp. 1409–1417, Jun. 2012.

[12] K. Y. Choi, J. H. Park, and D. H. Lee, "A new provably secure certificateless short signature scheme," *Comput. Math. Appl.*, vol. 61, no. 7, pp. 1760–1768, 2011.

[13] N. B. Gayathri, T. Gowri, R. R. V. Krishna Rao, and P. V. Reddy, "Efficient and secure pairing-free certificateless directed signature scheme," *J. King Saud Univ. Comput. Inf. Sci.*, to be published. doi: 10.1016/j.jksuci.2018.02.016.

[14] H. Yuan, F. Zhang, X. Huang, Y. Mu, and L. Zhang, "Certificateless threshold signature scheme from bilinear maps," *Inf. Sci.*, vol. 108, no. 23, pp. 4714–4728, Dec. 2010.

[15] H. Xiong, F. Li, and Z. Qin, "Certificateless threshold signature secure in the standard model," *Inf. Sci.*, vol. 237, pp. 73–81, Jul. 2013.

[16] L. Zhang, F. Zhang, and Q. Wu, "Delegation of signing rights using certificateless proxy signatures," *Inf. Sci.*, vol. 184, no. 1, pp. 298–309, Feb. 2012.

[17] S.-H. Seo, K. Y. Choi, J. Y. Hwang, and S. Kim, "Efficient certificateless proxy signature scheme with provable security," *Inf. Sci.*, vol. 188, pp. 322–337, Apr. 2012.

[18] D. He, Y. Chen, and J. Chen, "An efficient certificateless proxy signature scheme without pairing," *Math. Comput. Model.*, vol. 57, nos. 9–10, pp. 2510–2518, May 2013.

[19] Y. Lu and J. Li, "Provably secure certificateless proxy signature scheme in the standard model," *Theor. Comput. Sci.*, vol. 639, no. 1, pp. 42–59, Aug. 2016.

[20] H. Du and Q. Wen, "Certificateless proxy multi-signature," *Inf. Sci.*, vol. 276, no. 20, pp. 21–30, Aug. 2014.

[21] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2003, pp. 416–432.

[22] H. Xiong, Z. Guan, Z. Chen, and F. Li, "An efficient certificateless aggregate signature with constant pairing computations," *Inf. Sci.*, vol. 219, pp. 225–235, Jan. 2013.

[23] D. He, M. Tian, and J. Chen, "Insecurity of an efficient certificateless aggregate signature with constant pairing computations," *Inf. Sci.*, vol. 268, pp. 458–462, Jun. 2014.

[24] L. Cheng, Q. Wen, Z. Jin, H. Zhang, and L. Zhou, "Cryptanalysis and improvement of a certificateless aggregate signature scheme," *Inf. Sci.*, vol. 295, pp. 337–346, Feb. 2015.

[25] S.-J. Horng, S.-F. Tzeng, P.-H. Huang, X. Wang, T. Li, and M. K. Khan, "An efficient certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks," *Inf. Sci.*, vol. 317, pp. 48–66, Oct. 2015.

[26] P. Kumar, S. Kumari, V. Sharma, A. K. Sangaiah, J. Wei, and X. Li, "A certificateless aggregate signature scheme for healthcare wireless sensor network," *Sustain. Comput., Inform. Syst.*, vol. 18, pp. 80–89, Jun. 2018.

[27] J. Cui, J. Zhang, H. Zhong, R. Shi, and Y. Xu, "An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks," *Inf. Sci.*, vols. 451–452, pp. 1–15, Jul. 2018.

[28] H. Zhong, S. Han, J. Cui, J. Zhang, and Y. Xu, "Privacy-preserving authentication scheme with full aggregation in VANET," *Inf. Sci.*, vol. 476, pp. 211–221, Feb. 2019.

[29] X. Jia, D. He, Q. Liu, and K. K. R. Choo, "An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment," *Ad Hoc Netw.*, vol. 71, pp. 78–87, Mar. 2018.

[30] Y. Zhang, R. H. Deng, G. Han, and D. Zheng, "Secure smart health with privacy-aware aggregate authentication and access control in Internet of Things," *J. Netw. Comput. Appl.*, vol. 123, no. 12, pp. 89–100, Dec. 2018.

[31] M. Ma, D. He, N. Kumar, and J. Chen, "Certificateless searchable public key encryption scheme for mobile healthcare system," *Comput. Elect. Eng.*, vol. 65, pp. 413–424, Jan. 2018.

[32] J. Shen, Z. Gui, S. Ji, J. Shen, H. Tan, and T. Yi, "Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks," *J. Netw. Comput. Appl.*, vol. 106, pp. 117–123, Mar. 2018.

[33] M. E. S. Saeed, Q.-Y. Liu, G. Tian, B. Gao, and F. Li, "Remote authentication schemes for wireless body area networks based on the Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4926–4944, Dec. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8496749

[34] J. Shen, S. Chang, J. Shen, Q. Liu, and X. Sun, "A lightweight multi-layer authentication protocol for wireless body area networks," *Future Gener. Comput. Syst.*, vol. 78, pp. 956–963, Jan. 2018.

[35] L. Zhou, A. Fu, S. Yu, M. Su, and B. Kuang, "Data integrity verification of the outsourced big data in the cloud environment: A survey," *J. Netw. Comput. Appl.*, vol. 122, no. 15, Nov. 2018.

[36] Y. Miao, J. Weng, X. Liu, K. K. R. Choo, and H. Li, "Enabling verifiable multiple keywords search over encrypted cloud data," *Inf. Sci.*, vol. 465, pp. 21–37, Oct. 2018.

[37] Z. Yan, P. Wang, and W. Feng, "A novel scheme of anonymous authentication on trust in pervasive social networking," *Inf. Sci.*, vols. 445–446, pp. 79–96, Jun. 2018.

[38] J.-H. Zhang and J. Mao, "An efficient RSA-based certificateless signature scheme," *J. Syst. Softw.*, vol. 85, no. 3, pp. 638–642, Mar. 2012.

[39] S. Duan, "Certificateless undeniable signature scheme," *Inf. Sci.*, vol. 178, no. 3, pp. 742–755, Feb. 2008.

[40] W. Zhao and D. Ye, "Certificateless undeniable signatures from bilinear maps," *Inf. Sci.*, vol. 199, pp. 204–215, Sep. 2012.

[41] L. Zhang, Q. Wu, J. Domingo-Ferrerer, B. Qin, and P. Zeng, "Signatures in hierarchical certificateless cryptography: Efficient constructions and provable security," *Inf. Sci.*, vol. 272, pp. 223–237, Jul. 2014.

[42] K.-A. Shim, "Security models for certificateless signature schemes revisited," *Inf. Sci.*, vol. 296, pp. 315–321, Mar. 2015.

[43] M. H. Au, Y. Mu, J. Chen, D. S. Wong, J. K. Liu, and G. Yang, "Malicious KGC attacks in certificateless cryptography," in *Proc. 2nd ACM Symp. Inf., Comput. Commun. Secur.*, 2007, pp. 302–311.

[44] J. Huang and Q. Huang, "Black-box constructions of signature schemes in the bounded leakage setting," *Inf. Sci.*, vol. 423, pp. 313–325, Jan. 2018.

[45] M. Jakobsson, K. Sako, and K. R. Impaliazzo, "Designated verifier proofs and their applications," in *Proc. Eurocrypt*, in Lecture Notes in Computer Science, vol. 1070, New York, NY, USA: Springer-Verlag, 1996, pp. 142–154.

[46] S. Saeednia, S. Kreme, and O. Markotwich, "An efficient strong designated verifier signature scheme," in *Proc. ICISC*, in Lecture Notes in Computer Science, vol. 2971, New York, NY, USA: Springer-Verlag, 2003, pp. 40–54.

[47] X. Huang, W. Susilo, Y. Mu, and F. Zhang, "Certificateless designated verifier signature schemes," in *Proc. 20th Int. Conf. Adv. Inf. Netw. Appl.*, vol. 2, Apr. 2006, pp. 15–19.

[48] B. Yang, Z. Hu, and Z. Xiao, "Efficient certificateless strong designated verifier signature scheme," in *Proc. Int. Conf. Comput. Intell. Secur.*, vol. 1, Dec. 2009, pp. 432–436.

[49] Z. Xiao, B. Yang, and S. Li, "Certificateless strong designated verifier signature scheme," in *Proc. 2nd Int. Conf. E-Bus. Inf. Syst. Secur.*, May 2010, pp. 1–5.

[50] J. Zhang and J. Xie, "Breaking a certificateless strong designated verifier signature scheme," in *Proc. Int. Conf. Consum. Electron., Commun. Netw. (CECNet)*, Apr. 2011, pp. 130–133.

[51] S. K. H. Islam and G. P. Biswas, "Provably secure certificateless strong designated verifier signature scheme based on elliptic curve bilinear pairings," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 25, no. 1, pp. 51–61, 2013.

[52] T. Liu, X. Wang, and X. Ding, "Secure analysis and improvement of certificateless strong designated verifier signature scheme," *Comput. Sci.*, vol. 40, no. 7, pp. 126–128, 2013.

[53] Y. Chen, Y. Zhao, H. Xiong, and F. Yue, "A certificateless strong designated verifier signature scheme with non-delegatability," *Int. J. Netw. Security*, vol. 19, no. 4, pp. 573–582, 2017.

[54] H.-Y. Lin, "A new certificateless strong designated verifier signature scheme: Non-delegatable and SSA-KCA secure," *IEEE Access*, vol. 6, pp. 50765–50775, 2018.

[55] X. Hu, C. Ma, J. Wang, H. Xu, and W. Tan, "An undeniable strong DSVS scheme with no bilinear pairings," in *Proc. 9th Int. Congr. Image Signal Process., BioMed. Eng. Inform. (CISP-BMEI)*, Oct. 2017, pp. 1944–1948.

[56] X. Hu, W. Tan, H. Xu, J. Wang, and C. Ma, "Strong designated verifier signature schemes with undeniable property and their applications," *Secur. Commun. Netw.*, vol. 2017, Dec. 2017, Art. no. 7921782.

[57] A. Karati, S. H. Islam, and G. P. Biswas, "A pairing-free and provably secure certificateless signature scheme," *Inf. Sci.*, vol. 450, pp. 378–391, Jun. 2018.

[58] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *J. Cryptol.*, vol. 13, no. 3, pp. 361–396, 2000.

SHU HAN is currently pursuing the bachelor's degree with the School of Computer and Information Engineering, Zhejiang Gongshang University. Her current research interests include information and network security.

MANDE XIE is currently a Faculty Member with the School of Computer and Information Engineering, Zhejiang Gongshang University. His current research interests include mobile crowdsourcing, distributed computation, and network security.

BAILIN YANG is currently a Faculty Member with the School of Computer and Information Engineering, Zhejiang Gongshang University. His current research interests include mobile game, blockchain, mobile graphics, network security, and graphic computing.

RONGXING LU is currently a Faculty Member with the Faculty of Computer Science, University of New Brunswick. His current research interests include the Internet of Things (IoT)-big data security and privacy, privacy enhancing techniques, and applied cryptography. He is also the Vice-Chair (Publication) of the IEEE ComSoc's Communications and Information Security Technical Committee (CIS-TC).

HAIYONG BAO is currently a Faculty Member with the School of Computer and Information Engineering, Zhejiang Gongshang University. His current research interests include information security, data security, and privacy.

JIANHONG LIN is currently pursuing the Ph.D. degree with Zhejiang University. His current research interests include network security and security management. He is also the Chief Technology Officer with Zhejiang Ponshine Information Technology Company Ltd., Hangzhou, China.

HAI-BO HONG is currently an Associate Professor with the School of Computer and Information Engineering, Zhejiang Gongshang University. His current research interests include information security and cryptography.

MIAN-XUE GU is currently pursuing the master's degree with the School of Computer and Information Engineering, Zhejiang Gongshang University. His current research interests include network security, blockchain, and data privacy protection.

SONG HAN was a Visiting Professor with the School of Data Sciences, Eash China Normal University. He is currently a Faculty Member with the School of Computer and Information Engineering, Zhejiang Gongshang University. He is also with Zhejiang Ponshine Information Technology Company Ltd., Hangzhou, China. His current research interests include network security, blockchain, the Internet of Things, and data privacy protection.