Hindawi
Security and Communication Networks
Volume 2019, Article ID 5175076, 14 pages
https://doi.org/10.1155/2019/5175076

Research Article

Optimistic Fair Exchange in Cloud-Assisted Cyber-Physical Systems

Jun Wang,[1,2] **Feixiang Luo**,[1] **Zequan Zhou**,[1,2] **Xiling Luo**,[1,2] **and Zhen Wang**[1]

[1]*School of Electronic and Information Engineering, BeiHang University, Beijing, China*
[2]*BeiHang University HangZhou Innovation Institute, Hangzhou, China*

Correspondence should be addressed to Zequan Zhou; zy1702129@buaa.edu.cn

Received 16 April 2019; Revised 25 August 2019; Accepted 11 September 2019; Published 30 October 2019

Academic Editor: Clemente Galdi

Recently, optimistic fair exchange in electronic commerce (e-commerce) or mobile commerce (m-commerce) has made great progress. However, new technologies create large amounts of data and it is difficult to handle them. Fortunately, with the assistance of cloud computing and big data, optimistic fair exchange of digital items in cyber-physical systems (CPSes) can be efficiently managed. Optimistic fair exchange in cloud-assisted CPSes mainly focuses on online data exchange in e-commerce or online contracts signing. However, there exist new forms of risks in the uncertain network environment. To solve the above problems, we use a new technique called verifiably encrypted identity-based signature (VEIS) to construct optimistic fair exchange in cloud-assisted CPSes. VEIS is an encrypted signature, and we can check the validity of the underlying signature without decrypting it. We introduce a robust arbitration mechanism to guarantee fairness of the exchange, and even the trusted third party (TTP) cannot get the original signatures of the exchange parties. And the TTP in our protocol is offline, which greatly improves the efficiency. Besides, we show that our protocol is secure, fair, and practical.

1. Introduction

1.1. Background. In recent years, optimistic fair exchange in e-commerce and m-commerce has grown fast and information technology has been widely utilized. However, with the development of the information systems, the data are rapidly and continuously created from varieties of devices in the transaction, leaving big challenges for data storage and management. The new cloud-assisted cyber-physical systems (CPSes), which combine the big data, cloud computing, internet of things, and even artificial intelligence with industrial automation, can be used to deal with huge files and complex structures in the fair exchange.

The model of optimistic fair exchange in cloud-assisted CPSes is shown in Figure 1. In such systems, users can purchase the service or store their data in the cloud data service. Besides, they can also use their data to trade with others or sign contracts on digital files. For example, if Alice attempts to exchange her digital content with Bob in another cloud, she needs to perform optimistic fair exchange of signatures in her physical devices (e.g., desktop, laptop, or smartphone).

Although optimistic fair exchange in cloud-assisted CPSes plays a significant role in e-commerce or m-commerce with new capabilities, there still exist risks. Security and privacy issues continue to be a barrier for optimistic fair exchange due to the uncertain network environment. For uncertain environment of fair exchange, we consider two aspects: uncertain contents of the exchange and uncertain activities of the exchange [1, 2]. On the one hand, the exchange parties may not know what will be received before the exchange. On the other hand, no one can make sure the other party does honestly in the transaction. To solve the above problems, we proposed optimistic fair exchange, which is secure and practical in cloud-assisted CPSes.

1.2. Rated Works. Optimistic fair exchange is a common method in e-commerce or m-commerce, in which two parties (e.g., individuals, companies, or systems) exchange their signatures of digital goods fairly. In optimistic fair exchange protocols [3, 4], we usually consider three parties,

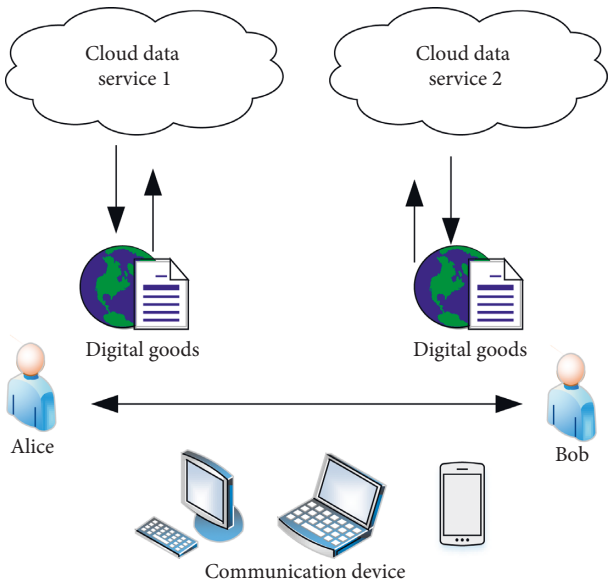

Figure 1: Optimistic fair exchange in cloud-assisted CPSes.

two exchange parties and a trusted third party (TTP, also called arbiter). Suppose Alice and Bob are parties involved in the protocol, and they need to exchange their signatures of contracts. In the protocol, Alice first sends a partial signature to Bob, and Bob then checks the validity of the partial signature. If it is invalid, Bob stops the protocol. Otherwise, Bob sends his full signature to Alice. Alice then checks whether Bob's full signature is valid. If it is valid, then Alice sends her full signature to Bob. If Bob does not receive Alice's full signature or receives an invalid signature, then Bob submits his full signature and Alice's partial signature to the TTP, the TTP first checks whether Bob's signature is valid. If it is valid, then the TTP converts the partial signature to the full signature and returns it to Bob. At the same time, the TTP returns Bob's full signature to Alice as well. Otherwise, the TTP does nothing.

There are other fair exchange protocols which use different technologies. In [5], the authors proposed a fair document exchange, which protected the anonymity of signers. Their another work [6] offered the protection of the private information and provided offline signature recovery. Fan et al. [7] used watermarking technology to construct fair content exchange in cloud computing; however, they did not give the full security proof. In [8], they showed optimistic fair exchange protocols which can be used in file sharing system with a high volume of data exchanged. However, all the above schemes have weaknesses. We compare these schemes with ours in the following aspects. The results are shown in Table 1.

(i) Identity-based cryptosystem: it does not need to exchange public and private keys and does not need key directories; thus, it has better efficiency than PKI-based cryptosystem.

(ii) Semitrusted third party (TTP): since malicious TTP may leak the information of the users, we need to use semi-TTP to replace fully trusted arbiter.

Table 1: Summary of fair exchange schemes.

	Identity-based cryptosystem	Semitrusted third party	Strong security
Zhang et al. [5]	No	No	No
Zhang et al. [6]	No	Yes	Yes
Fan et al. [7]	No	No	No
Kp and Lysyanskaya [8]	No	No	Yes
Gu et al. [9]	Yes	No	No
Zhang et al. [10]	Yes	No	No
Ours	Yes	Yes	Yes

(iii) Strong security: security is important for optimistic fair exchange; however, we find some schemes do not satisfy strong security since they do not give the attack model and security proof.

A kind of technique called verifiably encrypted signature (VES) can be used to construct optimistic fair exchange in cloud-assisted CPSes. VES is an encrypted signature, and we can check the validity without extracting the original signature. Some optimistic fair exchange protocols based on VES [9–13] have been proposed. However, most of them are based on Public Key Infrastructure (PKI), thus leading to the high cost in authenticating and managing the public keys. Fortunately, several identity-based VES schemes [9, 10, 12] have been proposed since identity-based cryptosystems can be used to reduce the cost in public key distribution and management. Most identity-based schemes are more efficient. With regard to security, most schemes consider only if an adversary cannot forge a valid original signature and a valid VES. However, they do not consider stronger security attributes: Can anyone forge a valid VES while the underlying signature is invalid? Meanwhile, many schemes do not give the attack model and security proof. Therefore, even though the schemes [9, 10, 12] are highly efficient, they are not very secure. We compare our scheme with the above schemes in four aspects in Table 2:

(i) Identity-based cryptosystem: it greatly reduces the cost in management of the public keys because it does not need to exchange public and private keys and does not need key directories.

(ii) Standard model: the security proof of a scheme does not use random oracle model.

(iii) Strong security: it means that the scheme satisfies three different security properties at the same time. The properties include opacity, unforgeability, and extractability. Meanwhile, there should be the attack model and security proof.

(iv) High efficiency: the lower the computational overhead involved in a scheme, the higher the efficiency. Therefore, high efficiency means less use of complex mathematical operations, including exponential, hash, and bilinear operations, while maintaining strong security properties. For a detailed computational overhead, refer to Table 3.

Table 2: Summary of VES schemes.

	Identity-based cryptosystem	Standard model	Strong security	High efficiency
Boneh et al. [11]	No	Yes	No	Yes
Gu [9]	Yes	No	No	Yes
Zhang et al. [10]	Yes	Yes	No	No
Lu et al. [12]	Yes	Yes	No	Yes
Nishimaki and Xagawa [13]	No	Yes	Yes	No
Shao and Gao [14]	No	No	Yes	No
Seo et al. [15]	No	Yes	Yes	No
Ours	Yes	Yes	Yes	Yes

Table 3: The efficiency of VES schemes.

	SKG	VESign	VEVerify	Adj	Ver
Boneh et al. [11]	$1E$	$3E+1H$	$3e+1H$	$1E$	$2e+1H$
Gu et al. [9]	$1H$	$2E+2H$	$3e+1H$	$1e+1H$	$2e+2H+1E$
Zhang et al. [10]	$2E$	$6E$	$5e$	$5E$	$4e$
Shao and Gao [14]	$2E$	$6E+2H$	$8E+2H$	$3E+1H$	$4E+2H$
Seo et al. [15]	$2E$	$6E+1H$	$4e+1H+1E$	$4E$	$3e$
Ours	$2E$	$6E$	$5e$	$1E$	$4e$

Motivated by the above works, we construct an optimistic fair exchange protocol with a new kind of VES scheme which combines an identity-based signature scheme and a PKI-based encryption scheme. In this way, we can achieve secure and efficient fair exchange in cloud-assisted CPSes.

1.3. Our Contributions. Technically, we use a new cryptographic technique called verifiably identity-based signature (VEIS) to construct the optimistic fair exchange protocol. The VEIS scheme which combines an identity-based signature scheme [16] and the ElGamal encryption scheme [17] allows us to exchange signatures in the cloud-assisted CPSes easily. Besides, we define the security properties according to the major requirements of optimistic fair exchange in cloud-assisted CPSes. The properties include opacity, unforgeability, and extractability. Opacity guarantees that no one can extract a valid signature from a VEIS without the private key. Unforgeability means that no one can forge a VEIS without the signing key of his identity. And extractability means that if a VEIS is valid, then a valid signature can be pulled out from it.

Then, we construct optimistic fair exchange in cloud-assisted CPSes with our VEIS scheme. In our protocol, two parties (Alice and Bob) may exchange their digital signatures in cloud with their physical devices. Then, Alice first generates her VEIS and sends it to Bob. Next, Bob sends his identity-based signature to Alice if Alice's VEIS is valid. Finally, Alice sends her identity-based signature to Bob. If cheating occurs during the exchange, the trusted third party (TTP) will get involved and ensure the fairness. Our protocol protects the privacy of the exchange parties since TTP will not get the original signatures of them. Besides, TTP only involves in the exchange when cheating occurs and it reduces the communication cost. The arbitration mechanism guarantees the fairness of the exchange, which means both parties obtain the other's signature or neither does when the protocol ends. Finally, we report the results of the simulation, which show that our protocol is efficient.

1.4. Outline. We organize the rest of this paper as follows. In Section 2, we give some relevant notions and show the building blocks of our scheme. In Section 3, we present the verifiably encrypted identity-based signature (VEIS) scheme which is used to construct our optimistic fair exchange in cloud-assisted CPSes. In Section 4, we construct the optimistic fair exchange protocol with offline TTP. Then, we discuss the security and fairness of our protocol. In Section 5, we show the results of the simulation. Finally, we conclude in Section 6.

2. The Main Text

2.1. Bilinear Maps and Complexity Assumption. We first briefly review the bilinear maps and complexity assumptions below.

2.1.1. Bilinear Maps. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p and g be a generator of $\mathbb{G}$, then we say that $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is an efficient and computable bilinear map if it satisfies the following properties, i.e., (1) *Bilinear*: $\forall a, b \in \mathbb{Z}_p$, we have $e(g^a, g^b) = e(g, g)^{ab}$. (2) *Non-degenerate*: $e(g, g) \neq 1$: clearly, the bilinearity implies that $\forall g_1, g_2, g_3 \in \mathbb{G}$, we have $e(g_1, g_3)e(g_2, g_3) = e(g_1 g_2, g_3)$.

2.1.2. Complexity Assumptions. CDH problem: given g, g^a, and g^b, compute g^{ab}.

If the probability that adversary $\mathbb{B}$ solves the CDH problem is at least ϵ, we have

$$\Pr\left[\mathbb{B}\left(g, g^a, g^b\right) = g^{ab}\right] \geq \varepsilon. \tag{1}$$

Then, CDH assumption is as follows.

Definition 1. The (t, ε)-CDH assumption holds if no adversary runs at most t times and has at least ϵ advantage in solving the CDH problem on $\mathbb{G}$.

The aggregate extraction problem: given g, g^a, g^b, g^β, g^δ, and $g^{ab+\beta\delta}$, compute g^{ab}.

If the probability that adversary $\mathbb{B}$ solves the aggregate extraction problem is at least ϵ, we have

$$\Pr\left[\mathbb{B}\left(g, g^a, g^b, g^\beta, g^\delta, g^{ab+\beta\delta}\right) = g^{ab}\right] \geq \varepsilon. \quad (2)$$

Then, aggregate extraction assumption is as follows.

Definition 2. The (t, ϵ)-aggregate extraction assumption holds if no adversary runs at most t times and has at least ϵ advantage in solving the aggregate extraction problem on $\mathbb{G}$.

2.2. Building Blocks. Our optimistic fair exchange protocol is mainly based on Paterson's identity-based signature (IBS) scheme [16] and ElGamal encryption scheme [17], which are described as follows.

2.2.1. Paterson's IBS Scheme. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p and g be a generator of $\mathbb{G}$. There exists an efficient and computable bilinear map $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$. All identities and messages will be assumed to be bit strings of length n_u and n_m, respectively. We can also let identities be bit strings of arbitrary length and n_u and n_m be the output length of collision-resistant hash functions, $H_u : \{0,1\} \longrightarrow \{0,1\}^{n_u}$ and $H_m : \{0,1\} \longrightarrow \{0,1\}^{n_m}$. The signature scheme is defined by the following algorithms.

Setup: a secret $\alpha \in \mathbb{Z}_p$ and generator g of $\mathbb{G}$ are chosen at random. Then, set the value $g_1 = g^\alpha$ and choose g_2 randomly in $\mathbb{G}$. Additionally, pick elements $u', m' \in \mathbb{G}$ randomly and vectors $\mathbf{U} = (u_i)$, $\mathbf{M} = (m_i)$ of length n_u and n_m, where u_i and m_i are randomly chosen from $\mathbb{G}$. The public parameters are MPK = $(\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, u', \mathbf{U}, m', \mathbf{M})$ and the master secret key is MSK = g_2^α.

Extract: let $\mathcal{U}$ be a bit string of length n_u representing an identity and let $\mathcal{U}[i]$ be the ith bit of $\mathcal{U}$. Define $\overline{U} \subset \{1, \ldots, n_u\}$ to be the set of indices i such that $\mathcal{U}[i] = 1$. A private key for identity $\mathcal{U}$ is generated as follows. First, choose a random $r_u \in \mathbb{Z}_p$. Then, compute

$$\mathrm{sk}_{\mathcal{U}} = d_{\mathcal{U}} = (d_1, d_2) = \left(g_2^\alpha \left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u}, g^{r_u}\right). \quad (3)$$

Sign: let $\mathcal{m}$ be a bit string of length n_m representing a message. Furthermore, let $\overline{M} \subset \{1, \ldots, n_m\}$ be a set of indices j such that $\mathcal{m}[j] = 1$.

A signature of $\mathcal{U}$ on $\mathcal{m}$ is constructed as follows: First, pick a random $r_m \in \mathbb{Z}_p$. Then, compute

$$\sigma = (\sigma_0, \sigma_1, \sigma_2) = \left(g_2^\alpha \left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u} \left(m' \prod_{j \in \overline{M}} m_j\right)^{r_m}, g^{r_u}, g^{r_m}\right). \quad (4)$$

Verify: suppose we wish to check if $\sigma = (\sigma_0, \sigma_1, \sigma_2)$ is a valid signature of an identity $\mathcal{U}$ on $\mathcal{m}$. The signature is accepted if it holds that

$$e(\sigma_0, g) = e(g_2, g_1) e\left(u' \prod_{i \in \overline{U}} u_i, \sigma_1\right) \cdot e\left(m' \prod_{j \in \overline{M}} m_j, \sigma_2\right). \quad (5)$$

2.2.2. The Modified Signature Scheme. In Paterson's IBS scheme, we find one element of the signature (σ_1) is a part of the private key (d_2) and it is not allowed in the strictest sense. Thus, we randomize the first two elements, including the secret key part of the original signature. Let Σ = (Setup, Extract, Sign, Verify) be Paterson's IBS scheme, and the new IBS scheme Σ' = (Setup′, Extract′, Sign′, Verify′) is as follows:

Setup′ and Extract′ are the same as Setup and Extract, respectively.

Sign′: choose random $r_u', r_m \in \mathbb{Z}_p$ and compute

$$\sigma = (\sigma_0, \sigma_1, \sigma_2) = \left(\sigma_0 \cdot \left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u'}, \sigma_1 \cdot g^{r_u'}, \sigma_2\right). \quad (6)$$

Verify′: we can check the validity of the signature by the following equation:

$$e(\sigma_0, g) \stackrel{?}{=} e(g_2, g_1) e\left(u' \prod_{i \in \overline{U}} u_i, \sigma_1\right) \cdot e\left(m' \prod_{j \in \overline{M}} m_j, \sigma_2\right). \quad (7)$$

If the equation holds, then the signature is valid. Otherwise, it is invalid.

The modified signature scheme is existentially unforgeable and the security proof is almost the same as stated in [16], we will not prove it again. We use the following theorem to show that.

Theorem 1. *The modified identity-based signature scheme is existentially unforgeable if the CDH assumption holds.*

2.2.3. The Encryption Scheme. In the ElGamal encryption scheme, KeyGen generates a pair of keys (SK, PK) = (α, g^α), where g is a generator of $\mathbb{G}$. Enc takes as input PK, a message $\mathcal{m}$, and outputs ciphertext $C = (C_1, C_2) = (\mathcal{m}(\mathrm{PK})^t, g^t)$, where t is randomly chosen from $\mathbb{G}$. Dec takes as input C, the secret key SK, and outputs the message $\mathcal{m} = C_1 / C_2^\alpha = \mathcal{m}$.

3. Verifiably Encrypted Identity-Based Signature

Verifiable encrypted identity-based signature (VEIS) is an encrypted identity-based signature, allowing its validity to be checked without decryption. We encrypt the identity-based signature with the public key encryption scheme. In fact, our

scheme is a weak version of the identity-based verifiably encrypted signature scheme as stated in [18]; thus, we call it verifiably encrypted identity-based signature (VEIS). Informally, the VEIS system works as follows. A private key generator (PKG) generates private keys for the signers, and a key generator (KG) generates keys pairs for the adjudicator. A signer generates a signature on a message with a signing key associated with his identity, then encrypts it with the adjudicator's public key, and obtains a VEIS. A verifier can publicly check whether the VEIS is valid. The adjudicator can decrypt VEIS with his private key and get the original signature. We now present our definition of VEIS, as well as several properties.

3.1. Modelling VEIS. Formally, a VEIS scheme VEIS = (Setup, SKG, VESign, VEVer, Adj, Ver) is defined as follows.

Setup: it takes as input a security parameter 1^λ and then outputs two pairs of keys, (MSK, MPK) for PKG and $(\mathrm{SK}_T, \mathrm{PK}_T)$ for the adjudicator.

SKG: PKG inputs 1^λ, a master secret key MSK, and a signer's identity $\mathcal{U}$ and outputs a private key $\mathrm{sk}_{\mathcal{U}} \longleftarrow \mathrm{SKG}(1^\lambda, \mathrm{MSK}, \mathcal{U})$ for the signer.

VESign: it takes as input a signing key $\mathrm{sk}_{\mathcal{U}}$ of identity $\mathcal{U}$ and a message m; a signer first generates a signature σ and then he encrypts it with the public key PK_T and outputs a VEIS $\omega \longleftarrow \mathrm{VESign}(m, \mathrm{sk}_{\mathcal{U}}, \mathrm{PK}_T)$.

VEVer: deterministic algorithm VEVer takes as input a message m, a VEIS ω, and verification keys $\mathcal{U}$, PK_T, and MPK and then outputs a bit $b \longleftarrow \mathrm{VEVer}(m, \omega, \mathrm{MPK}, \mathrm{PK}_T, \mathcal{U})$, where $b \in \{0, 1\}$. If $b = 1$, then VEIS is valid. Otherwise, it is invalid.

Adj: the adjudicator takes as input ω and SK_T, decrypts VEIS, and then obtains a signature $\sigma \longleftarrow \mathrm{Adj}(\omega, \mathrm{SK}_T)$.

Ver: a verifier can check whether the signature is valid. That is, $\mathrm{Ver}(m, \sigma, \mathrm{MPK}, \mathcal{U}) = b$, where $b \in \{0, 1\}$. If $b = 1$, the signature is valid. Otherwise, the signature is invalid.

3.2. Security Definitions. In [11], Boneh et al. proposed two security definitions of verifiably encrypted signature (VES), i.e., unforgeability and opacity. Unforgeability means that no one can forge a VES without the signing key. Opacity guarantees that no one can extract a valid signature from a VES without the adjudicator's private key. In fact, we find that the VES scheme in [11] missed an important property, i.e., extractability. Extractability means that a valid signature can be extracted from a VES if it passes the check. And we require that extractability must hold even when the master public key and encrypted signatures are maliciously generated.

We now formalize three security definitions of VEIS, i.e., unforgeability, opacity, and extractability. They are all defined in games which are played by a challenger and an adversary. Besides, we define the following oracles:

(i) Signing key oracle: the adversary can make a query for a private key associated with an identity $\mathcal{U}$ of a signer, the challenger first gets the master key, runs SKG, and then responds with the private key $\mathrm{sk}_{\mathcal{U}}$.

(ii) VEIS oracle: the adversary can make a query for a VEIS on message m of an identity $\mathcal{U}$; the challenger first gets a private key of $\mathcal{U}$, runs VESign, and responds with a VEIS ω.

(iii) Adjudication oracle: the adversary can make a query for a signature by providing a message m, a VEIS ω, and an identity $\mathcal{U}$. The challenger first gets the decryption key, runs Adj, and responds with a signature σ.

Then, the security definitions are as follows.

Definition 3. Unforgeability is defined by the game $\mathrm{Game}_{\mathrm{Forge}}(\lambda)$. The involved parties in the game are a challenger and an adversary $\mathbb{A}$.

(i) Setup: the challenger sets up the system, generates the public parameters, and sends them to $\mathbb{A}$.

(ii) Query: $\mathbb{A}$ submits an identity $\mathcal{U}$ to **Signing Key Oracle** and obtains a signing key $\mathrm{sk}_{\mathcal{U}}$. $\mathbb{A}$ can make at most q_1 queries for signing keys. $\mathbb{A}$ submits an identity $\mathcal{U}$, a message m to **VEIS Oracle**, and obtains a VEIS ω. $\mathbb{A}$ can make at most q_2 queries for VEIS. $\mathbb{A}$ submits a message m, a VEIS ω, and an identity $\mathcal{U}$ to **Adjudication Oracle** and obtains a signature σ. $\mathbb{A}$ can make at most q_3 queries for signatures.

(iii) Forge: finally, $\mathbb{A}$ submits a tuple $(m^*, \omega^*, \mathcal{U}^*)$ to the challenger, where the adversary never queried **Signing Key Oracle** at $\mathcal{U}^*$, **VEIS Oracle** at $(m^*, \mathcal{U}^*)$, or **Adjudication Oracle** at $(m^*, \omega^*, \mathcal{U}^*)$. If $\mathrm{VEVer}(m^*, \omega^*, \mathrm{MPK}, \mathrm{PK}_T, \mathcal{U}^*) = 1$, $\mathbb{A}$ wins in the above game.

A VEIS scheme is unforgeable if for any probabilistic polynomial time (PPT) adversary $\mathbb{A}$, the probability that $\mathbb{A}$ wins in the above game is negligible.

Definition 4. pacity is defined by the game $\mathrm{Game}_{\mathrm{Opac}}(\lambda)$ as follows. Similarly, a challenger and an adversary get involved in the game.

(i) Setup: the challenger sets up the system, generates the public parameters, and sends them to $\mathbb{A}$.

(ii) Query: $\mathbb{A}$ submits an identity $\mathcal{U}$ to **Signing Key Oracle** and obtains a signing key $\mathrm{sk}_{\mathcal{U}}$. $\mathbb{A}$ can make at most q_1 queries for signing keys. $\mathbb{A}$ submits an identity $\mathcal{U}$, a message m to **VEIS Oracle**, and obtains a VEIS ω. $\mathbb{A}$ can make at most q_2 queries for VEIS. $\mathbb{A}$ submits a message m, a VEIS ω, and an identity $\mathcal{U}$ to **Adjudication Oracle** and obtains a signature σ. $\mathbb{A}$ can make at most q_3 queries for signatures.

(iii) Forge: finally, $\mathbb{A}$ submits a tuple $(m^*, \sigma^*, \mathcal{U}^*)$ to the challenger, where the adversary never queried **Signing Key Oracle** at $\mathcal{U}^*$, or **Adjudication Oracle** at $(m^*, \omega^*, \mathcal{U}^*)$, and $\mathbb{A}$ can get ω^* from **VEIS**

Oracle. If σ^* is a valid signature on m^* of identity $\mathcal{U}^*$, $\mathbb{A}$ wins in the above game.

A VEIS scheme is opaque if for any PPT adversary $\mathbb{A}$, the probability that $\mathbb{A}$ wins in the above game is negligible.

Definition 5. Extractability is defined by the game $\text{Game}_{\text{Extr}}(\lambda)$, which involves a challenger and an adversary $\mathbb{A}$.

(i) Setup: the challenger sets up the system, generates the public parameters, and sends them to $\mathbb{A}$.

(ii) Query: $\mathbb{A}$ submits an identity $\mathcal{U}$ to **Signing Key Oracle** and obtains a signing key $\text{sk}_{\mathcal{U}}$. $\mathbb{A}$ submits a message m, a VEIS ω, and an identity $\mathcal{U}$ to **Adjudication Oracle** and obtains a signature σ.

(iii) Forge: $\mathbb{A}$ submits a tuple $(m^*, \omega^*, \mathcal{U}^*, \text{MPK}^*)$ to the challenger.

(iv) Extract: the challenger runs Adj and gets σ^*. If we have $\text{VEVer}(m^*, \omega^*, \text{MPK}^*, \text{PK}_T, \mathcal{U}^*) = 1$ and $\text{Ver}(\sigma^*, m^*, \text{MPK}^*, \mathcal{U}^*) = 0$, which means that the signature σ^* is invalid, then $\mathbb{A}$ wins in the above game.

A VEIS scheme is extractable if for all PPT adversaries $\mathbb{A}$, the probability that adversaries wins in the above game is negligible.

We require that if a VEIS scheme satisfies the unforgeability, opacity, and extractability, then it is secure.

3.3. The Proposed VEIS Scheme. We now present the concrete VEIS scheme. A VEIS scheme consists of following algorithms.

Setup: it takes as input a security parameter 1^λ and generates the system parameters. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p, g be a generator of $\mathbb{G}$, and e be a bilinear map; that is, $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$. An identity for a signer is denoted by $\mathcal{U}$. We assume that the length of the identity is n_u. If it is not, then we use a hash function $H_u : \{0,1\} \longrightarrow \{0,1\}^{n_u}$ to obtain n_u-length identity. Then, choose n_u-length vectors $\mathbf{U} = (u_i)$ and n_m-length vector $\mathbf{M} = (m_j)$ at random, where $u_i, m_j \in \mathbb{G}$. Besides, pick random $u', m' \in \mathbb{G}$. Choose random $\alpha_1 \in \mathbb{Z}_p$ as the secret value. Compute $g_1 = g^{\alpha_1}$. Choose $g_2 \in \mathbb{G}$ randomly. Then, set the master secret key $\text{MSK} = g_2^{\alpha_1}$ and the master public key $\text{MPK} = (\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, u', \mathbf{U}, m', \mathbf{M})$. Besides, choose random $\alpha_T \in \mathbb{Z}_p$ and then set the private key for adjudication as $\text{SK}_T = \alpha_T$, and the public key is $\text{PK}_T = g^{\alpha_T}$.

SKG: it takes as input an identity of a signer, $\mathcal{U}$, and the master secret key MSK and outputs a private key for signing as follows. First, let $\overline{U} \subset \{1, \ldots, n_u\}$ be the set of indices i such that $\mathcal{U}[i] = 1$, where $\mathcal{U}[i]$ is the ith bit of $\mathcal{U}$. Then, choose a random $r_u \in \mathbb{Z}_p$ and compute

$$\text{sk}_{\mathcal{U}} = d_{\mathcal{U}} = \left(g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i \right)^{r_u}, g^{r_u} \right). \tag{8}$$

VESign: it takes as input a signing key $d_{\mathcal{U}}$ and a message m and generates a signature as follows. Let $\overline{M} \subset \{1, \ldots, n_m\}$ be the set of indices j such that $m[j] = 1$, where $m[j]$ is the jth bit of m. Then, choose a random $r_u', r_m \in \mathbb{Z}_p$ and construct a signature by computing

$$\begin{aligned}
\sigma_0 &= g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i \right)^{r_u} \left(u' \prod_{i \in \overline{U}} u_i \right)^{r_u'} \cdot \left(m' \prod_{j \in \overline{M}} m_j \right)^{r_m}, \\
\sigma_1 &= g^{r_u} g^{r_u'}, \\
\sigma_2 &= g^{r_m}, \\
\sigma &= (\sigma_0, \sigma_1, \sigma_2).
\end{aligned} \tag{9}$$

Then, we construct a VEIS as follows. In fact, since σ_1 and σ_2 are independent with the message and identity $\mathcal{U}_1$, we only need to encrypt σ_0. Choose a random $t \in \mathbb{Z}_p$ and construct the VEIS by computing

$$\begin{aligned}
\omega_1 &= (\text{PK}_T)^t \sigma_0, \\
&= (\text{PK}_T)^t g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i \right)^{r_u + r_u'} \left(m' \prod_{j \in \overline{M}} m_j \right)^{r_m}, \\
\omega_2 &= g^t, \\
\omega_3 &= \sigma_1 = g^{r_u + r_u'}, \\
\omega_4 &= \sigma_2 = g^{r_m}, \\
\omega &= (\omega_1, \omega_2, \omega_3, \omega_4).
\end{aligned} \tag{10}$$

VEVer: if the following equation holds:

$$\begin{aligned}
e(\omega_1, g) = e(\text{PK}_T, \omega_2) e(g_2, g_1) e \\
\cdot \left(\left(u' \prod_{i \in \overline{U}} u_i \right), \omega_3 \right) e \left(\left(m' \prod_{j \in \overline{M}} m_j \right), \omega_4 \right),
\end{aligned} \tag{11}$$

then, the VEIS is valid and the verifier outputs 1. Otherwise, he outputs 0.

Adj: it takes as input ω and $\text{SK}_T = \alpha_T$ and outputs a signature by computing

$$\begin{aligned}
\sigma_0 &= \frac{\omega_1}{\omega_2^{\alpha_T}}, \\
\sigma_1 &= \omega_3, \\
\sigma_2 &= \omega_4, \\
\sigma &= (\sigma_0, \sigma_1, \sigma_2).
\end{aligned} \tag{12}$$

Ver: it checks whether $\sigma = (\sigma_0, \sigma_1, \sigma_2)$ is valid. If the equation

$$e(\sigma_0, g) = e(g_2, g_1)e\left(u' \prod_{i \in \overline{U}} u_i, \sigma_1\right)e\left(\left(m' \prod_{j \in \overline{M}} m_j\right), \sigma_2\right), \tag{13}$$

holds, then the signature is valid and Ver outputs 1. Otherwise, Ver outputs 0 to show σ is invalid.

Our scheme is secure in the standard model, and the security proof will be shown in the next section.

4. Optimistic Fair Exchange in Cloud-Assisted CPSes

In this section, we construct optimistic fair exchange in cloud-assisted CPSes based on the proposed VEIS scheme. The system model is shown in Figure 2. Two parties (known as Alice and Bob) may want to exchange their signatures on digital goods in cloud. They first get their private keys from a private key generator. Then, they perform our optimistic fair exchange protocol and get the signature of each other fairly. If there exist disputes, they can ask the TTP for arbitration.

4.1. Optimistic Fair Exchange Protocol. It is more straightforward to design the fair exchange protocol with online TTP than optimistic fair exchange (fair exchange protocol with offline TTP). However, optimistic fair exchange is more efficient since it greatly reduces the workload on the TTP. In the optimistic fair exchange protocol, the TTP only involves in the exchange when someone attempts to cheat. And TTP in our protocol will not get the original signature of the parties in the exchange. Without loss of generality, we assume Alice and Bob have agreed on two files m_A and m_B, and Alice is the protocol initiator. The concrete protocol is as follows:

(i) Initialization: take as input security parameter 1^λ, run $Setup(1^\lambda)$, and output master public key $\text{MPK} = (\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, u', \mathbf{U}, m', \mathbf{M})$ and the master secret key $\text{MSK} = g_2^{\alpha_1}$. Besides, the TTP chooses random $\alpha_T \in \mathbb{Z}_p$, and then he sets a key pair for adjudication as $(\text{SK}_T, \text{PK}_T) = (\alpha_T, g^{\alpha_T})$. Similarly, A and B choose random $\alpha_A, \alpha_B \in \mathbb{Z}_p$, and then set their key pairs as $(\text{SK}_A, \text{PK}_A) = (\alpha_A, g^{\alpha_A})$ and $(\text{SK}_B, \text{PK}_B) = (\alpha_B, g^{\alpha_B})$, respectively.

(ii) Registration: a user can register with his identity u and obtain a signing key. Let u be a bit string of length n_u and let $u[i]$ be the ith bit of u. Define $\overline{U} \subset \{1, \ldots, n_u\}$ to be the set of indices i such that $u[i] = 1$. The signing key for identity u is computed as follows:

$$\text{sk}_u = d_u = (d_1, d_2) = \left(g_2^{\alpha_1}\left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u}, g^{r_u}\right). \tag{14}$$

(iii) Exchange: let u_A and u_B be identities of A and B, respectively. Then Alice and Bob will obtain their signing keys $d_{u_A} = (d_{A,1}, d_{A,2})$ and $d_{u_B} = (d_{B,1}, d_{B,2})$, respectively. The time t_1 denotes the maximum time Alice can wait, and the time t_2 denotes the longest the Bob can wait. Then, the optimistic fair exchange of identity-based signatures is as follows:

(1) Alice first generates her identity-based signature (on m_A) as

$$\begin{aligned}\sigma_A &= \left(\sigma_{A,0}, \sigma_{A,1}, \sigma_{A,2}\right) \\ &= \left(g_2^{\alpha_1}\left(u' \prod_{i \in \overline{U}_A} u_i\right)^{r_{u_1} + r'_{u1}}\left(m' \prod_{j \in \overline{M}_A} m_j\right)^{r_{m_1}}, \right. \\ &\quad \left. \cdot\, g^{r_{u_1} + r'_{u_1}}, g^{r_{m_1}}\right).\end{aligned} \tag{15}$$

Then, she encrypts it with the new public key $\text{PK} = \text{PK}_T \cdot \text{PK}_B = g^{\alpha_T + \alpha_B}$ and obtains

$$\begin{aligned}\omega_{A,1} &= (PK)^t \sigma_{A,0}, \\ \omega_{A,2} &= g^t, \\ \omega_{A,3} &= \sigma_{A,1} = g^{r_{u_1} + r'_{u1}}, \\ \omega_{A,4} &= \sigma_{A,2} = g^{r_{m_1}}, \\ \omega_A &= \left(\omega_{A,1}, \omega_{A,2}, \omega_{A,3}, \omega_{A,4}\right).\end{aligned} \tag{16}$$

Then, she sends $\omega_A \| m_B$ to Bob.

(2) Bob quits if he waited more than t_2. Otherwise, after receiving $\omega_A \| m_B$ from Alice, Bob checks whether the following equation holds:

$$\begin{aligned}e\left(\omega_{A,1}, g\right) \stackrel{?}{=}\ & e\left(\text{PK}, \omega_{A,2}\right)e\left(g_2, g_1\right)e \\ & \cdot \left(\left(u' \prod_{i \in \overline{U}_A} u_i\right), \omega_{A,3}\right)e \\ & \cdot \left(\left(m' \prod_{j \in \overline{M}_A} m_j\right), \omega_{A,4}\right).\end{aligned} \tag{17}$$

If the equation holds, then ω_A is valid and it implies that the identity-based signature σ_A is valid as well. Bob generates his identity-based signature (on m_B), encrypts it with the new public key $\text{PK} = \text{PK}_T \cdot \text{PK}_A = g^{\alpha_T + \alpha_A}$, and obtains

$$\begin{aligned}\omega_{B,1} &= (\text{PK})^t \sigma_{B,0}, \\ \omega_{B,2} &= g^t, \\ \omega_{B,3} &= \sigma_{B,1} = g^{r_{u_1} + r'_{u1}}, \\ \omega_{B,4} &= \sigma_{B,2} = g^{r_{m_1}}, \\ \omega_B &= \left(\omega_{B,1}, \omega_{B,2}, \omega_{B,3}, \omega_{B,4}\right).\end{aligned} \tag{18}$$

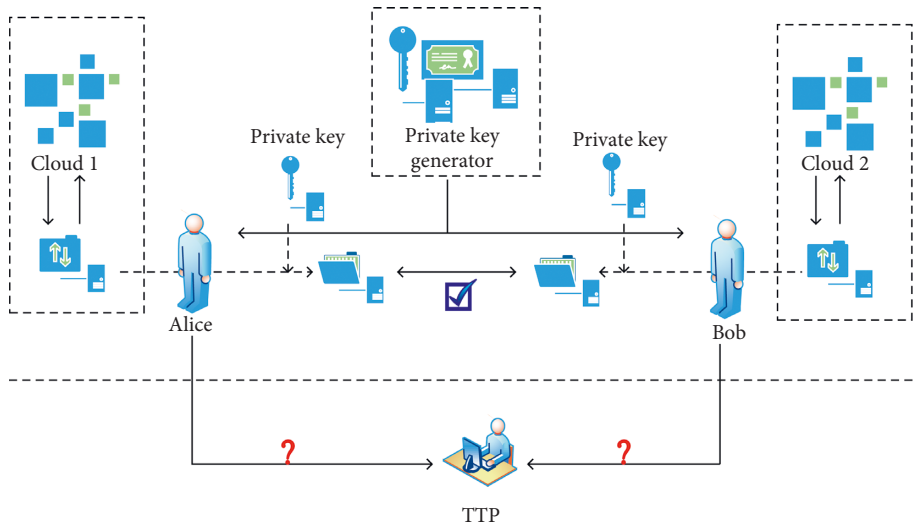

Figure 2: System model.

Then Bob sends $\omega_B \| m_A$ to Alice.

(3) Alice invokes Protocol *Abort* and then quits if she waited more than t_1. Otherwise, within the time t_1, Alice receives $\omega_B \| m_A$ from Bob and checks whether it is valid by the following equation:

$$e(\omega_{B,1}, g) \stackrel{?}{=} e(\mathrm{PK}, \omega_{B,2}) e(g_2, g_1) e \cdot \left(\left(u' \prod_{i \in \overline{U}_B} u_i \right), \omega_{B,3} \right) e \cdot \left(\left(m' \prod_{j \in \overline{M}_B} m_j \right), \omega_{B,4} \right). \tag{19}$$

If the equation holds, Bob's signature is valid and Alice sends her identity-based signature σ_A to Bob. Otherwise, Alice invokes Protocol *Abort* and then quits.

(4) Bob invokes Protocol *Resolve* if he waited more than t_2. Otherwise, within the time t_2, Bob receives σ_A from Alice and checks it. If it is valid, Bob sends his identity-based signature σ_B to Alice. Otherwise, Bob invokes Protocol *Resolve* and then quits.

(5) Alice invokes Protocol *Resolve* if she waited more than t_1. Otherwise, within the time t_1, Alice receives σ_B from Bob and checks whether it is valid. If Bob's signature is valid, the protocol ends successfully. Otherwise, Alice invokes Protocol *Resolve* and then quits.

(i) Abort protocol:

(a) Alice sends an abort command [Alice, Bob, abort] to TTP.

(b) The TTP checks the record if there have a command of resolve from Bob.

(c) If not, the TTP sends acceptation message to Alice and stores the abort command into record. Otherwise, the TTP sends Bob's signature ω'_B.

(ii) Resolve protocol: Alice and Bob can use the same *Resolve* protocol. For convenience, we use Bob as an example to describe the protocol.

(a) Bob encrypts σ_B with the public key $\mathrm{PK}_A = g^{\alpha_A}$ and obtains

$$\begin{aligned} \omega'_{B,1} &= (\mathrm{PK}_A)^{t'} \sigma_{B,0}, \\ \omega'_{B,2} &= g^{t'}, \\ \omega'_{B,3} &= \sigma_{B,1} = g^{r_{u_2} + r'_{u_2}}, \\ \omega'_{B,4} &= \sigma_{B,2} = g^{r_{m_2}}, \\ \omega'_B &= (\omega'_{B,1}, \omega'_{B,2}, \omega'_{B,3}, \omega'_{B,4}). \end{aligned} \tag{20}$$

Then he submits a resolve command [Alice, Bob, $\omega_A \| m_B, \omega'_B$, resolve] to TTP.

(b) The TTP checks the record if there is a command from Alice to ask abort the exchange. If it exists, the TTP sends a rejection message.

(c) If not, the TTP first checks whether $\omega_A = (\omega_{A,1}, \omega_{A,2}, \omega_{A,3}, \omega_{A,4})$ and $\omega'_B = (\omega'_{B,1}, \omega'_{B,2}, \omega'_{B,3}, \omega'_{B,4})$ are valid by the following equations:

$$\begin{aligned} e(\omega_{A,1}, g) &\stackrel{?}{=} e(g_2, g_1) e \left(\left(u' \prod_{i \in \overline{U}_A} u_i \right), \omega_{A,3} \right) e \cdot (\mathrm{PK}, \omega_{A,2}) e \left(\left(m' \prod_{j \in \overline{M}_A} m_j \right), \omega_{A,4} \right), \\ e(\omega'_{B,1}, g) &\stackrel{?}{=} e(g_2, g_1) e \left(\left(u' \prod_{i \in \overline{U}_B} u_i \right), \omega'_{B,3} \right) e \cdot (\mathrm{PK}_A, \omega'_{B,2}) e \left(\left(m' \prod_{j \in \overline{M}_B} m_j \right), \omega'_{B,4} \right). \end{aligned} \tag{21}$$

If either of the equations does not hold, then the TTP refuses the request from Bob. Otherwise, the TTP

decrypts ω_A with its private key and obtains a new VEIS by computing

$$\omega'_{A,1} = \frac{\omega_{A,1}}{\omega_{A,2}^{\alpha_T}} = (\mathrm{PK}_B)^t g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}_A} u_i\right)^{r_{u_1}+r'_{u1}} \cdot \left(m' \prod_{j \in \overline{M}_A} m_j\right)^{r_{m_1}},$$
$$\omega'_{A,2} = \omega_{A,2} = g^t,$$
$$\omega_{A,3} = \omega_{A,3} = g^{r_{u_1}+r'_{u1}},$$
$$\omega'_{A,4} = \omega_{A,4} = g^{r_{m_1}}. \tag{22}$$

Next TTP returns $\omega'_A = (\omega'_{A,1}, \omega'_{A,2}, \omega'_{A,3}, \omega'_{A,4})$ to Bob and sends ω'_B to Alice. Meanwhile, the TTP stores this command into record.

(d) Bob can decrypt ω'_A with his private key and obtains Alice's signature σ_A by computing

$$\sigma_{A,0} = \frac{\omega'_{A,1}}{\omega'^{\alpha_B}_{A,2}} = g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}_A} u_i\right)^{r_{u_1}+r'_{u1}} \left(m' \prod_{j \in \overline{M}_A} m_j\right)^{r_{m_1}},$$
$$\sigma_{A,1} = \omega'_{A,3} = g^{r_{u_1}+r'_{u1}},$$
$$\sigma_{A,2} = \omega'_{A,3} = g^{r_{m_1}},$$
$$\sigma_A = (\sigma_{A,0}, \sigma_{A,1}, \sigma_{A,2}). \tag{23}$$

4.2. Security. In fact, the security of our protocol can be reduced to the security of the VEIS scheme as stated in [9]. Therefore, we mainly consider the following questions. Can anyone forge a valid original signature? Can anyone forge a valid VEIS? Given a VEIS, can anyone extract a valid signature from it? And there is an extra question which is missed in [9, 10]: Can anyone forge a valid VEIS while the underlying signature is invalid? The above questions are the basis of the security properties. Therefore, we will show that our scheme satisfies unforgeability, opacity, and extractability by the following theorems.

Theorem 2. *Our VEIS scheme is unforgeable.*

Proof. In fact, if $\mathbb{A}$ forges a valid VEIS ω^* (on m^*) of identity $\mathcal{U}^*$, he breaks the unforgeability of our VEIS scheme, and we can construct an adversary $\mathbb{B}$ that breaks the existential unforgeability of the underlying signature scheme. Therefore, the unforgeability of our VEIS scheme can be reduced to the underlying IBS scheme, which is slightly different from Paterson's IBS scheme. We now prove Theorem 2 by the following two lemmas.

Lemma 1. *Our VEIS scheme is unforgeable if the underlying signature scheme is existentially unforgeable.*

Proof. If there exists an adversary $\mathbb{A}$ that forges a tuple $(m^*, \omega^*, \mathcal{U}^*)$ with a nonnegligible probability ϵ and it holds that $\mathrm{VEVer}(m^*, \omega^*, \mathrm{MPK}, \mathrm{PK}_T, \mathcal{U}^*) = 1$, then $\mathbb{A}$ breaks the unforgeability of the VEIS scheme. And we can construct an adversary $\mathbb{B}$ that breaks the existential unforgeability of the underlying signature scheme.

We assume that $\mathbb{A}$ and $\mathbb{B}$ play the game $\mathrm{Game}_{\mathrm{Forge}}(\lambda)$, and $\mathbb{B}$ simulates the challenger and tries to break the underlying signature scheme. We now show how to construct $\mathbb{B}$.

Setup: $\mathbb{B}$ runs algorithm Setup and gets the master secret key $\mathrm{MSK} = g_2^{\alpha_1}$, the master public key $\mathrm{MPK} = (\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, u', \mathbf{U}, m', \mathbf{M})$, the adjudicator's public key $\mathrm{PK}_T = g^{\alpha_T}$, and adjudicator's private key $\mathrm{SK}_T = \alpha_T$. Then he sends MPK and PK_T to $\mathbb{A}$, which are indistinguishable from the real ones.

Query: $\mathbb{B}$ simulates oracles as follows:

(a) Simulation of signing key oracle: when $\mathbb{A}$ submits an identity $\mathcal{U}$ of a signer and requests a private key, $\mathbb{B}$ chooses r_u at random and computes

$$\mathrm{sk}_{\mathcal{U}} = d_{\mathcal{U}} = \left(g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u}, g^{r_u}\right), \tag{24}$$

and then he sends it to $\mathbb{A}$.

(b) Simulation of VEIS oracle: when $\mathbb{A}$ submits a message m and an identity $\mathcal{U}$ and requests a VEIS, $\mathbb{B}$ first gets a private key of $\mathcal{U}$ as he does in **Simulation of Signing Key Oracle**, and he generates the signature as follows:

$$\sigma_0 = g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u} \left(u' \prod_{i \in \overline{U}} u_i\right)^{r'_u} \left(m' \prod_{j \in \overline{M}} m_j\right)^{r_m},$$
$$\sigma_1 = g^{r_u} g^{r'_u},$$
$$\sigma_2 = g^{r_m},$$
$$\sigma = (\sigma_0, \sigma_1, \sigma_2). \tag{25}$$

Then, he encrypts the signature with PK_T by

$$\omega_1 = (PK_T)^t \sigma_0 = (PK_T)^t g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u+r'_u} \left(m' \prod_{j \in \overline{M}} m_j\right)^{r_m},$$
$$\omega_2 = g^t,$$
$$\omega_3 = \sigma_1 = g^{r_u+r'_u},$$
$$\omega_4 = \sigma_2 = g^{r_m}. \tag{26}$$

Finally, he sends $\omega = (\omega_1, \omega_2, \omega_3, \omega_4)$ to $\mathbb{A}$.

(c) Simulation of adjudication oracle: when $\mathbb{A}$ submits a message m, a VEIS $\omega = (\omega_1, \omega_2, \omega_3, \omega_4)$, and an identity $\mathcal{U}$ and requests a signature, $\mathbb{B}$ decrypts the VEIS as follows:

$$\begin{aligned}\sigma_0 &= \frac{\omega_1}{\omega_2^{\alpha_T}},\\ \sigma_1 &= \omega_3,\\ \sigma_2 &= \omega_4.\end{aligned} \tag{27}$$

Then, he sends the signature $\sigma = (\sigma_0, \sigma_1, \sigma_2)$ to $\mathbb{A}$.

The simulation above is perfect since $\mathbb{B}$ does it honestly.

Forge: finally, $\mathbb{A}$ submits a tuple $(m^*, \omega^*, \mathcal{U}^*)$ to $\mathbb{B}$, where the adversary has never queried **Signing Key Oracle** at $\mathcal{U}^*$, **VEIS Oracle** at $(m^*, \mathcal{U}^*)$, or **Adjudication Oracle** at $(m^*, \omega^*, \mathcal{U}^*)$. If $\mathrm{VEVer}(m^*, \omega^*, \mathrm{MPK}, \mathrm{PK}_T, \mathcal{U}^*) = 1$, then we have $\mathrm{Ver}(m^*, \sigma^*, \mathrm{MPK}, \mathcal{U}^*) = 1$ (it implies the extractability of VEIS and we will prove this later), and $\mathbb{B}$ can decrypt the VEIS and obtain a valid signature as follows:

$$\begin{aligned}\sigma_0^* &= \frac{\omega_1^*}{\omega_2^{*\alpha_T}},\\ \sigma_1^* &= \omega_3^*,\\ \sigma_2^* &= \omega_4^*,\\ \sigma^* &= \left(\sigma_0^*, \sigma_1^*, \sigma_2^*\right).\end{aligned} \tag{28}$$

Thus, $\mathbb{B}$ forges a valid signature $\sigma^* = (\sigma_0^*, \sigma_1^*, \sigma_2^*)$ with a nonnegligible probability ϵ. This completes the proof.

Lemma 2. *The modified signature scheme is existentially unforgeable.*

The proof is almost the same as stated in [14], and we use Theorem 1 to show that.

Theorem 3. *Our VEIS scheme is opaque if the aggregate extraction assumption holds, and the underlying signature scheme is existentially unforgeable.*

Proof. Suppose adversary $\mathbb{A}$ breaks the opacity of the VEIS scheme with nonnegligible probability ϵ, then we can construct an adversary $\mathbb{B}$ that solves the aggregate extraction problem. $\mathbb{A}$ and $\mathbb{B}$ play the game $\mathrm{Game}_{\mathrm{Opac}}(\lambda)$, and $\mathbb{B}$ simulates the challenger.

The aggregate extraction problem: given $g, g^a, g^b, g^\beta, g^\delta$, and $g^{ab+\beta\delta}$, compute g^{ab}.

We now construct $\mathbb{B}$ as follows:

Setup. $\mathbb{B}$ chooses n_u-length vectors $\mathbf{U} = (u_i)$, and n_m-length vector $\mathbf{M} = (m_j)$ at random, where u_i, $m_j \in \mathbb{G}$. Besides, pick random u', $m' \in \mathbb{G}$. Then he sets $g_1 = g^a$, $g_2 = g^b$, and the master public key $\mathrm{MPK} = (\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, u', \mathbf{U}, m', \mathbf{M})$, and public key $\mathrm{PK}_T = g^\beta$. Then he sends MPK and PK_T to $\mathbb{A}$, which are indistinguishable from the real ones. Although $\mathbb{B}$ does not know $\mathrm{MSK} = g_2^a$ and $\mathrm{SK}_T = \beta$, we can still complete the simulation by playing some tricks.

$\mathbb{B}$ sets $l_u = 2(q_1 + q_2)$ and $l_m = 2q_2$ ($\mathbb{A}$ can query at most q_1 times for private keys and q_2 times for VEISs). Then, $\mathbb{B}$ chooses x', z', n_u-length vector $X = (x_i)$, and n_m-length vector $Z = (z_j)$ at random, where x' and x_i are random values in $\{0, \ldots, l_u\}$ and z' and z_j are random values in $\{0, \ldots, l_m\}$. Besides, $\mathbb{B}$ picks y', w', n_u-length vector $Y = (y_i)$, and n_m-length vector $W = (w_j)$, where y', y_i, w', and w_j are random elements in $\mathbb{Z}_p$. Then, $\mathbb{B}$ sets $u' = g_2^{-l_u k_1 + x'} g^{y'}$, $u_i = g_2^{x_i} g^{y_i}$, $m' = g_2^{-l_m k_2 + z'} g^{w'}$, and $m_j = g_2^{z_j} g^{w_j}$, where $0 \le k_1 \le n_u$ and $0 \le k_2 \le n_m$.

Then, define the following functions for easy analysis:

$$\begin{aligned}F_1(\mathcal{U}) &= -l_u k_1 + x' + \sum_{i \in \overline{U}} x_i,\\ K_1(\mathcal{U}) &= y' + \sum_{i \in \overline{U}} y_i,\\ F_2(m) &= -l_m k_2 + z' + \sum_{j \in \overline{M}} z_j,\\ K_2(m) &= w' + \sum_{j \in \overline{M}} w_j.\end{aligned} \tag{29}$$

Query. $\mathbb{B}$ simulates oracles as follows:

(a) Simulation of signing key oracle: when $\mathbb{A}$ submits an identity $\mathcal{U}$ of a signer and requests a private key, $\mathbb{B}$ constructs it as follows:

$$\begin{aligned}d_{\mathcal{U}} &= (d_1, d_2)\\ &= \left(g_1^{\left(-K_1(\mathcal{U})/F_1(\mathcal{U})\right)}\left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u}, g_1^{\left(-1/F_1(\mathcal{U})\right)} g^{r_u}\right).\end{aligned} \tag{30}$$

Writing $\widetilde{r}_u = r_u - (a/F_1(\mathcal{U}))$, the private key can be simplified as

$$\begin{aligned}d_1 &= g_1^{\left(-K_1(\mathcal{U})/F_1(\mathcal{U})\right)}\left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u}\\ &= g_1^{\left(-K_1(\mathcal{U})/F_1(\mathcal{U})\right)}\left(g_2^{F_1(\mathcal{U})} g^{K_1(\mathcal{U})}\right)^{r_u}\\ &= g_2^a\left(g_2^{F_1(\mathcal{U})} g^{K_1(\mathcal{U})}\right)^{-\left(a/F_1(\mathcal{U})\right)}\left(g_2^{F_1(\mathcal{U})} g^{K_1(\mathcal{U})}\right)^{r_u}\\ &= g_2^a\left(u' \prod_{i \in \overline{U}} u_i\right)^{r_u - \left(a/F_1(\mathcal{U})\right)}\\ &= g_2^a\left(u' \prod_{i \in \overline{U}} u_i\right)^{\widetilde{r}_u},\\ d_2 &= g_1^{-\left(1/F_1(\mathcal{U})\right)} g^{r_u} = g^{r_u - \left(a/F_1(\mathcal{U})\right)} = g^{\widetilde{r}_u}.\end{aligned} \tag{31}$$

Therefore, the private keys generated by $\mathbb{B}$ are indistinguishable from the real ones. Then, $\mathbb{B}$ sends the signing key $d_{\mathcal{U}} = (d_1, d_2)$ to $\mathbb{A}$.

(b) Simulation of VEIS oracle: when $\mathbb{A}$ submits a message m_ℓ and an identity $\mathcal{U}_\ell$ and requests a VEIS, $\mathbb{B}$ can use a list Query List $:= = \varnothing$ to respond. $\mathbb{B}$ initializes the list Query List $:= = \varnothing$ and chooses the random index $\ell^* \in \{1, \ldots, q_2\}$ to guess from which VEIS $\mathbb{A}$ extracts a signature in the list. And $\mathbb{A}$ has not queried for a signing key at $\mathcal{U}_{\ell^*}$. If $\ell \neq \ell^*$, then $\mathbb{B}$ chooses random r_u, r_u', and r_m in $\mathbb{Z}_p$ and constructs VEIS as follows. First, $\mathbb{B}$ computes

$$\begin{aligned}\sigma_{0,\ell} &= g_1^{(-K_2(m_\ell)/F_2(m_\ell))}\left(u'\prod_{i\in\overline{U}}u_i\right)^{r_u}\left(u'\prod_{i\in\overline{U}}u_i\right)^{r_u'}\\ &\quad\cdot\left(m'\prod_{j\in\overline{M}}m_j\right)^{r_m}\\ &= g_2^{a}\left(u'\prod_{i\in\overline{U}}u_i\right)^{r_u+r_u'}\left(m'\prod_{j\in\overline{M}}m_j\right)^{\widetilde{r}_m},\\ \sigma_{1,\ell} &= g^{r_u}g^{r_u'} = g^{r_u+r_u'},\\ \sigma_{2,\ell} &= g_1^{(-1/F_2(m))}g^{r_m} = g^{\widetilde{r}_m},\\ \sigma_\ell &= \left(\sigma_{0,\ell}, \sigma_{1,\ell}, \sigma_{2,\ell}\right),\end{aligned} \tag{32}$$

where $\widetilde{r}_m = r_m - (a/F_2(m_\ell))$. Then, he chooses random t and encrypts the signature with PK_T as follows:

$$\begin{aligned}\omega_{1,\ell} &= (PK_T)^t\sigma_0\\ &= (PK_T)^t g_2^a\left(u'\prod_{i\in\overline{U}}u_i\right)^{r_u+r_u'}\left(m'\prod_{j\in\overline{M}}m_j\right)^{\widetilde{r}_m},\\ \omega_{2,\ell} &= g^t,\\ \omega_{3,\ell} &= \sigma_{1,\ell} = g^{r_u+r_u'},\\ \omega_{4,\ell} &= \sigma_{2,\ell} = g^{\widetilde{r}_m}.\end{aligned} \tag{33}$$

If $\ell = \ell^*$, then $\mathbb{B}$ chooses random r_u, r_u', r_m and sets

$$\begin{aligned}\omega_{1,\ell^*} &= g^{ab+\beta\delta}\left(u'\prod_{i\in\overline{U}}u_i\right)^{r_u+r_u'}\left(m'\prod_{j\in\overline{M}}m_j\right)^{r_m}\\ &= \left(g^\beta\right)^\delta g_2^a\left(u'\prod_{i\in\overline{U}}u_i\right)^{r_u+r_u'}\left(m'\prod_{j\in\overline{M}}m_j\right)^{r_m},\\ \omega_{2,\ell^*} &= g^\delta,\\ \omega_{3,\ell^*} &= g^{r_u+r_u'},\\ \omega_{4,\ell^*} &= g^{r_m}.\end{aligned} \tag{34}$$

Finally, he sends the VEIS $\omega_\ell = (\omega_{1,\ell}, \omega_{2,\ell}, \omega_{3,\ell}, \omega_{4,\ell})$ and $\omega_{\ell^*} = (\omega_{1,\ell^*}, \omega_{2,\ell^*}, \omega_{3,\ell^*}, \omega_{4,\ell^*})$ to $\mathbb{A}$ and stores the tuple $(m_\ell, \sigma_\ell, \omega_\ell, \mathcal{U}_\ell)$, $(m_{\ell^*}, \sigma_{\ell^*}, \omega_{\ell^*}, \mathcal{U}_{\ell^*})$ in QueryList. If one of $F_1(\mathcal{U}_\ell) = 0$, $F_2(m_\ell) = 0$, $F_1(\mathcal{U}_{\ell^*}) \neq 0$, and $F_2(m_{\ell^*}) \neq 0$ holds, $\mathbb{B}$ stops the game (if one of them holds, $\mathbb{B}$ cannot solve his problem). We denote this event by S_1. Otherwise, the simulation is perfect.

(c) Simulation of adjudication oracle: in this phase, $\mathbb{A}$ is not allowed to make a query on $(m_{\ell^*}, \omega_{\ell^*})$. When $\mathbb{A}$ submits a message $m_{\ell'}$, a VEIS $\omega_{\ell'} = (\omega_{1,\ell'}, \omega_{2,\ell'}, \omega_{3,\ell'}, \omega_{4,\ell'})$, and an identity $\mathcal{U}_{\ell'}$ and requests a signature, $\mathbb{B}$ first checks whether the tuple $(m_{\ell'}, \omega_{\ell'}, \mathcal{U}_{\ell'})$ is in the list QueryList; if it is not in QueryList, then the VEIS is invalid and $\mathbb{B}$ returns $\perp$ (if the VEIS is valid, then $\mathbb{A}$ forges a valid VEIS, and this contradicts the unforgeability of our VEIS scheme). If the tuple is in the list QueryList and $\ell' = \ell$, then $\mathbb{B}$ finds out the tuple $(\mathcal{U}_\ell, m_\ell, \sigma_\ell, \omega_\ell)$ and returns σ_ℓ to $\mathbb{A}$. The simulation above is perfect if $\mathbb{B}$ has not aborted.

Forge: finally, $\mathbb{A}$ outputs a valid signature extracted from the VEIS; that is, $\sigma_{\ell^*} = (\sigma_{0,\ell^*}, \sigma_{1,\ell^*}, \sigma_{2,\ell^*})$ (on message m_{ℓ^*}) of identity $\mathcal{U}_{\ell^*}$ with a nonnegligible probability ϵ. That means $\mathbb{B}$ correctly guesses from which VEIS $\mathbb{A}$ extracts the identity-based signature, and we denote this event by S_2. Then, $\mathbb{B}$ solves his problem by computing

$$g^{ab} = \frac{\sigma_{0,\ell^*}}{\left(\sigma_{1,\ell^*}\right)^{K_1(\mathcal{U}_{\ell^*})}\left(\sigma_{2,\ell^*}\right)^{K_2(m_{\ell^*})}}. \tag{35}$$

Then, the probability that $\mathbb{B}$ has not aborted in the above game is as follows:

$$\Pr[S_1 \wedge S_2] = \Pr[S_1]\Pr[S_2]. \tag{36}$$

The probability that $\mathbb{B}$ correctly guesses the index ℓ^* is $1/q_2$, and we deduce $\Pr[S_1] \geq 1/(16q_2(q_1+q_2)(n_u+1)(n_m+1))$ since we use proof skills in [16]. Thus, we have

$$\begin{aligned}\Pr[S_1 \wedge S_2] &\geq \frac{1}{q_2}\cdot\frac{1}{16q_2(q_1+q_2)(n_u+1)(n_m+1)}\\ &= \frac{1}{16q_2^2(q_1+q_2)(n_u+1)(n_m+1)}.\end{aligned} \tag{37}$$

And the probability that $\mathbb{B}$ solves the aggregate extraction problem is at least $\varepsilon/(16q_2^2(q_1+q_2)(n_u+1)(n_m+1))$, which is nonnegligible.

This completes the proof of opacity.

Theorem 4. *Our VEIS scheme is extractable.*

Proof. The challenger plays the game $\mathrm{Game}_{\mathrm{Extr}}(\lambda)$ with an adversary $\mathbb{A}$ as follows:

Setup: the challenger sets up the system, generates key pairs $(\mathrm{MSK}, \mathrm{MPK})$, $(\mathrm{PK}_T, \mathrm{SK}_T)$, and sends the public

keys $\mathrm{MPK} = (\mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, u', \mathbf{U}, m', \mathbf{M})$ and $\mathrm{PK}_T = g^{\alpha_T}$ to $\mathbb{A}$.

Query: when $\mathbb{A}$ makes queries, the challenger answers as follows:

(a) Signing key oracle: $\mathbb{A}$ submits an identity $\mathcal{U}$, then the challenger chooses random $r_{\mathcal{U}} \in \mathbb{Z}_p$ and computes

$$\mathrm{sk}_{\mathcal{U}} = d_{\mathcal{U}} = \left(g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i \right)^{r_u}, g^{r_u} \right). \tag{38}$$

Then, he returns it to $\mathbb{A}$.

(b) Adjudication oracle: $\mathbb{A}$ submits a message m, a VEIS $\omega = (\omega_1, \omega_2, \omega_3, \omega_4)$, and an identity $\mathcal{U}$, then the challenger first decrypts the VEIS and obtains

$$\begin{aligned} \sigma_0 &= \frac{\omega_1}{\omega_2^{\alpha_T}} = g_2^{\alpha_1} \left(u' \prod_{i \in \overline{U}} u_i \right)^{r_u + r_u'} \left(m' \prod_{j \in \overline{M}} m_j \right)^{r_m}, \\ \sigma_1 &= \omega_3 = g^{r_u + r_u'}, \\ \sigma_2 &= \omega_4 = g^{r_m}, \\ \sigma &= (\sigma_0, \sigma_1, \sigma_2). \end{aligned} \tag{39}$$

Then, he returns $\sigma = (\sigma_0, \sigma_1, \sigma_2)$ to $\mathbb{A}$.

Forge: finally, $\mathbb{A}$ outputs a VEIS $\omega^* = (\omega_1^*, \omega_2^*, \omega_3^*, \omega_4^*)$ of the challenge identity $\mathcal{U}^*$ and sends $(m^*, \omega^*, \mathcal{U}^*, \mathrm{MPK}^*)$ to the challenger.

Extract: if the given VEIS $\omega^* = (\omega_1^*, \omega_2^*, \omega_3^*, \omega_4^*)$ passes the check, then we have

$$\begin{aligned} e(\omega_1^*, g) = e(\mathrm{PK}_T, \omega_2^*) e(g_2, g_1) e\left(\left(u' \prod_{i \in \overline{U}} u_i \right), \omega_3^* \right) e \\ \cdot \left(\left(m' \prod_{j \in \overline{M}} m_j \right), \omega_4^* \right). \end{aligned} \tag{40}$$

Then, a signature can be extracted by

$$\begin{aligned} \sigma_0^* &= \frac{\omega_1^*}{\omega_2^{*\alpha_T}}, \\ \sigma_1^* &= \omega_3^*, \\ \sigma_2^* &= \omega_4^*, \\ \sigma^* &= (\sigma_0^*, \sigma_1^*, \sigma_2^*). \end{aligned} \tag{41}$$

And we have

$$\begin{aligned} e(\sigma_0^*, g) &= e\left(\frac{\omega_1^*}{\omega_2^{*\alpha_T}}, g \right) = e(\omega_1^*, g) e(\omega_2^{*\alpha_T}, g)^{-1} \\ &= e(\mathrm{PK}_T, \omega_2^*) e(g_2, g_1) e\left(\left(u' \prod_{i \in \overline{U}} u_i \right), \omega_3^* \right) e \\ &\quad \cdot \left(\left(m' \prod_{j \in \overline{M}} m_j \right), \omega_4^* \right) e(\omega_2^*, g^{\alpha_T})^{-1} \\ &= e(g_2, g_1) e\left(\left(m' \prod_{j \in \overline{M}} m_j \right), \sigma_2^* \right) e\left(u' \prod_{i \in \overline{U}} u_i, \sigma_1^* \right). \end{aligned} \tag{42}$$

It implies that if $\mathrm{VEVer}(m^*, \omega^*, \mathrm{MPK}^*, \mathrm{PK}_T, \mathcal{U}^*) = 1$ holds, then $\mathrm{Ver}(m^*, \sigma^*, \mathrm{MPK}^*, \mathcal{U}^*) = 1$ always holds as well. Thus, our VEIS scheme is extractable.

4.3. Fairness. To show our protocol satisfies fairness, we consider the cases where Alice or Bob cheats in the protocol.

(i) Case 1: Alice has two chances to perform the optimistic fair exchange protocol dishonestly. In step 1, if Alice sends an invalid VEIS ω_A to Bob, then Bob will check whether the verification equation holds. Since ω_A is invalid, the equation does not hold. Bob detects that ω_A is invalid and stops the protocol. In step 3, after receiving Bob's signature, Alice does not send her signature σ_A to Bob or sends an invalid signature to Bob. However, Bob can submit ω_A and ω_B to the TTP and ask for arbitration. If both ω_A and ω_B are valid, the TTP decrypts ω_A and returns ω_A' to Bob. Then, Bob decrypts ω_A' with his private key and obtains Alice's signature σ_A. Therefore, Alice gets no benefits if she cheats in the protocol.

(ii) Case 2: Bob also has two chances to perform the optimistic fair exchange protocol dishonestly. In step 1, after receiving the valid signature ω_A from Alice, Bob stops the protocol and asks for arbitration without providing his VEIS ω_B or providing an invalid VEIS. However, TTP will detect this and return nothing to Bob. In step 2, after receiving the valid signature σ_A from Alice, Bob sends an invalid signature to Alice. However, Alice will detect this and refuse to send σ_A to Bob. Therefore, Bob also gets no benefits if he cheats in the protocol.

Besides, the TTP will not get useful information of the exchange parties.

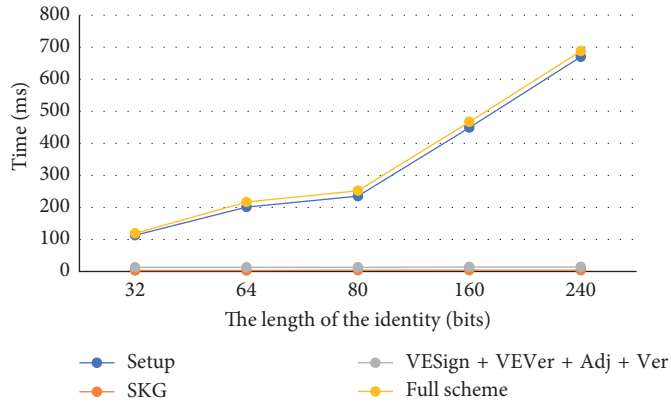

Figure 3: Running time of VEIS.

5. Simulation

In this section, we report several simulations and the results show that our VEIS scheme and optimistic fair exchange protocol are practical. We conducted experiments with Pairing Based Cryptography (PBC, http://crypto.stanford.edu/pbc/) library. The experiments were compiled in C programming language and carried out on Inter(R) Core(TM) Quad CPU Core i7-6700K @ 4.00 GHz. The security parameter was set to be 160, and the length of an element in group $\mathbb{G}$, $\mathbb{G}_T$ was 512 bits. The efficiency of our VEIS scheme mainly depends on the length of users' identities, which was chosen from 32 bits to 240 bits to evaluate the performance of our VEIS scheme and optimistic fair exchange protocol.

Before performing our experiments, we first compare several VES schemes by theoretical analysis. The results are shown in Table 3.

Let E, H, and e denote exponent operation time, operation time of hash function, and pairing operation time, and all the other light-weight operations such as additions and multiplications are omitted. We can find the efficiency of our scheme is comparable with other efficient schemes. The VES schemes in [9, 11] are more efficient; however, Boneh's scheme is PKI-based and Gu's scheme is not quite secure. Although VES schemes in [14, 15] are more secure, too many hash operations still reduce the efficiency of the schemes. Our VEIS scheme overcomes their weaknesses and is still efficient. The simulation results are shown in the following papers.

In Figure 3, we show the running time of our VEIS scheme with different length of identities (n_u). We do not consider the length of files (n_m) since we have the similar results when only changing n_m. The length of identities is chosen from 32 bits to 240 bits; that is, n_u = 32, 64, 80, 160, 240. And the full time ranges from 120 ms to 688 ms. The algorithm Setup takes roughly the same processing time, which is the biggest part of full operation time. To the contrary, SKG costs about 3 ms, and the rest algorithms costs about 16 ms for different length of identities. Therefore, once the system is set up, it is very efficient for users to generate their VEISs in cloud. Since the private key generator must distribute all users' keys, we have to evaluate whether it is suitable for a large number of users who take activities in cloud-assisted CPSes. The length of identity is set to be 160 bits, and Figure 4 shows the results. For cloud data services which have 100000 users, our scheme only takes about 350 seconds to generate their signing keys.

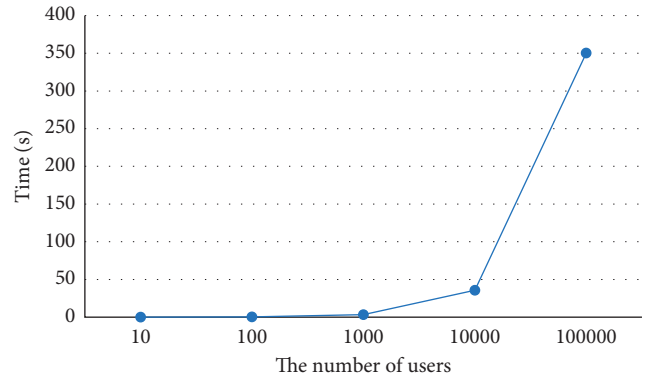

Figure 4: SKG time for different users.

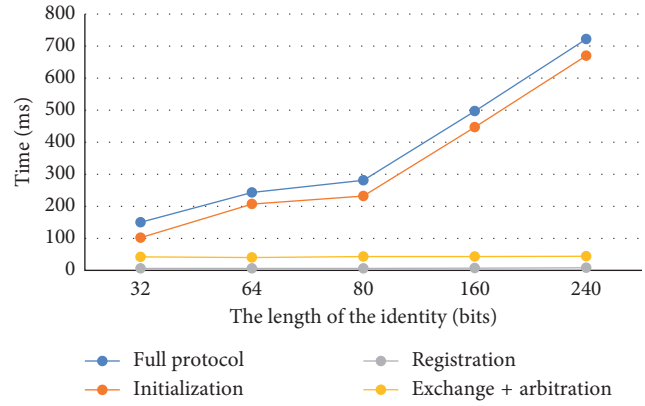

Figure 5: Running time of optimistic fair exchange.

In Figure 5, we show the running time of optimistic fair exchange with different length of identities. In the experiments, we do not consider the communication time which is hard to evaluate due to some subjective factors. The full protocol takes less than 1 second when the length of the identity is set to be 240 bits, and it is quite acceptable. Initialization takes a little more time than other phases; however, we only need to perform it once in the system.

When two users exchange their signatures of digital files in cloud, they only need to take 50 or 60 ms to finish the transaction. The above results show that our protocol is efficient and practical in cloud-assisted CPSes.

6. Conclusion

Optimistic fair exchange in cloud-assisted CPSes is a kind of protocol in which two parties can exchange their signatures of digital content in cloud. In this paper, we constructed such a protocol with a new cryptographic technique, i.e., verifiably encrypted identity-based signature (VEIS), which is based on Paterson's IBS scheme and ElGamal encryption scheme. Then, we showed that our protocol is secure with the strictest security proof. Additionally, we discussed fairness of our protocol, and the results showed that both two parties obtain each other's signature or neither of them get anything useful in the protocol. Finally, the results of the simulation showed that our protocol is practical.

Data Availability

These data relate to the personal privacy of some authors. Therefore, the data used to support the findings of this study are available from the corresponding author upon request via mail (ZY1702129@buaa.edu.cn).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. U17733115).

References

[1] C. H. Wang and C. M. Wang, *Novel Design of Fair Exchange Protocol for Semi-trusted Server and its Application in Cloud Environment, Information Security*, IEEE, Piscataway, NJ, USA, 2016.

[2] Y. Wang, Q. Wu, D. S. Wong, B. Qin, J. Mao, and Y. Ding, "Provably secure robust optimistic fair exchange of distributed signatures," *Future Generation Computer Systems*, vol. 62, pp. 29–39, 2016.

[3] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures (extended abstract)," in *Lecture Notes in Computer Science*, K. Nyberg, Ed., pp. 591–606, Springer, Berlin, Germany, 1998.

[4] F. Bao, R. H. Deng, and W. Mao, "Efficient and practical fair exchange protocols with offline TTP," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 77–85, Oakland, CA, USA, May 1998.

[5] N. Zhang, Q. Shi, and M. Merabti, "An efficient protocol for anonymous and fair document exchange," *Computer Networks*, vol. 41, no. 1, pp. 19–28, 2003.

[6] N. Zhang, Q. Shi, M. Merabti, and R. Askwith, "Autonomous mobile agent based fair exchange," *Computer Networks*, vol. 46, no. 6, pp. 751–770, 2004.

[7] C. I. Fan, W. S. Juang, and M. T. Chen, *Efficient Fair Content Exchange in Cloud Computing, Computer Symposium*, IEEE, Piscataway, NJ, USA, 2010.

[8] A. Kp and A. Lysyanskaya, "Usable optimistic fair exchange," *Computer Networks*, vol. 56, no. 1, pp. 50–63, 2012.

[9] C. X. Gu, Y. F. Zhu, and Y. J. Zhang, "An id-based optimistic fair signature exchange protocol from pairings," in *Lecture Notes in Computer Science*, D. Y. Hao, Ed., vol. 3802, pp. 9–16, Springer, Berlin, Germany, 2005.

[10] L. Zhang, Q. Wu, and B. Qin, "Identity-based optimistic fair exchange in the standard model," *Security and Communication Networks*, vol. 6, no. 8, pp. 1010–1020, 2013.

[11] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Lecture Notes in Computer Science*, E. Biham, Ed., vol. 2656, pp. 416–432, Springer, Berlin, Germany, 2003.

[12] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters, "Sequential aggregate signatures, multisignatures, and verifiably encrypted signatures without random oracles," *Journal of Cryptology*, vol. 26, no. 2, pp. 340–373, 2013.

[13] R. Nishimaki and K. Xagawa, "Verifiably encrypted signatures with short keys based on the decisional linear problem and obfuscation for encrypted VES," *Designs, Codes and Cryptography*, vol. 77, no. 1, pp. 61–98, 2015.

[14] Z. Shao and Y. Gao, "Practical verifiably encrypted signatures based on discrete logarithms," *Security and Communication Networks*, vol. 9, no. 18, 2016.

[15] J. H. Seo, K. Emura, K. Xagawa, and K. Yoneyama, "Accumulable optimistic fair exchange from verifiably encrypted homomorphic signatures," *International Journal of Information Security*, vol. 17, no. 2, pp. 193–220, 2018.

[16] K. G. Paterson and J. C. N. Schuldt, "Efficient identity-based signatures secure in the standard model," in *Lecture Notes in Computer Science*, vol. 4058, pp. 207–222, Springer, Berlin, Germany, 2006.

[17] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *Advances in Cryptology*, vol. 196, pp. 10–18, 2000.

[18] D. Galindo, J. Herranz, and E. Kiltz, "On the generic construction of identity-based signatures with additional properties," in *Lecture Notes in Computer Science*, X. Lai and K. Chen, Eds., vol. 4284, pp. 178–193, Springer, Berlin, Germany, 2006.