Strong Designated Verifier Signature Scheme with Undeniability and Strong Unforgeability in the Standard Model

Xiaodong Yang *, Guilan Chen, Ting Li, Rui Liu, Meiding Wang and Caifen Wang

Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China; chenglpaper@163.com (G.C.); lt1358382@163.com (T.L.); Liurui0556@163.com (R.L.); wmnding95@163.com (M.W.); wangcfen@126.com (C.W.)

* Correspondence: y200888@163.com

Received: 22 April 2019; Accepted: 14 May 2019; Published: 19 May 2019

Abstract: Strong designated verifier signature can provide an efficient way to protect the identity privacy of the signer and the integrity of the data transmitted over the public channel. These characteristics make it very useful in outsourcing computing, electronic voting, electronic bidding, electronic auction and other fields. However, most strong designated verifier signature schemes are unable to identify the real signature generator when the signer and the designated verifier dispute a signature. In addition, the existing strong designated verifier signature schemes in the standard model rarely satisfy strong unforgeability, and thus cannot prevent the attacker from forging a valid signature on any previously signed message. Therefore, designing a strong designated verifier signature scheme without random oracles that satisfies strong unforgeability and undeniability is very attractive in both practice and theory. Motivated by these concerns, we design the first undeniable strong designated verifier signature scheme without random oracles, in which the arbiter can independently perform the judgment procedure to prove whether a controversial signature is generated by the signer or the designated verifier. Under standard assumptions, the scheme is proved to be strongly unforgeable in standard model. Furthermore, it not only achieves non-transferability and privacy of the signer’s identity but also satisfies the undeniable property of traditional digital signature schemes. Performance analysis results show that the length of the signer’s private key, the designated verifier’s private key and signature length are 40 bits, 40 bits and 384 bits, respectively. Compared with the related schemes, the proposed scheme has higher performance in signature length, private key size and computational overhead. Finally, we show how to apply it to implement outsourcing computation in cloud computing.

Keywords: strong designated verifier signature; undeniability; strong unforgeability; standard model

1. Introduction

Digital signature is a very important information security technology, which can realize data integrity, non-repudiation, identity authentication and other functions. It plays an important role in network security communication [1], e-commerce [2], e-government [3] and other systems [4–6]. To deal with specific application scenarios, some digital signature schemes with special properties have been proposed. Among them, designated verifier signature (DVS) [7] is a significant variant of digital signature. In DVS, the signer is allowed to designate a verifier to confirm the authenticity of a signature, but the designated verifier is unable to convince anyone that the signature was generated by the real signer. The reason is that the simulated signature produced by the designated verifier is computationally indistinguishable from the original signature created by the signer for the same
message. This feature of DVS is called non-transferability, which is very useful in the fields of electronic voting, electronic tendering and software copyright [8,9]. To avoid the signer’s identity information being leaked, Jakobsson et al. [7] introduced the concept of strong designated verifier signature (SDVS). In SDVS, the validation of a signature must require the designated verifier’s private key, and any third party cannot determine the real creator of the signature. That is to say, only the designated verifier knows the real identity of the signer. Thus, SDVS enhances the privacy of the signer’s identity (PSI) and can be applied to some new fields [10]. For example, in cognitive computation [11], an intelligent robot authenticates the identity of its owner, but it must protect the owner’s identity information.

However, in a SDVS scheme, any third party does not know who generated the signature when the signer and the designated verifier dispute a signature. In this scenario, the undeniability property is very essential for SDVS. There are a few SDVS schemes with undeniability, and they were proved to be secure in the random oracle model [12–14]. Unfortunately, Canetti et al. [15] showed that the cryptographic scheme in the random oracle model may be insecure when the random oracle is instantiated by a concrete hash function. Therefore, it is of practical significance to study the SDVS scheme without random oracles.

Most existing SDVS schemes in the standard model only possess existential unforgeability [16–18]. Namely, an adversary can easily obtain a new legal signature of the same message by modifying an existing message-signature pair. Strong unforgeability can prevent the above-mentioned modification and protect the integrity of a signature [19]. A SDVS scheme is said to be strongly unforgeable if it satisfies existential unforgeability and the adversary cannot produce a legal signature of a message that has previously been signed. Although strong unforgeability has already been considered in several SDVS schemes [20], none of them has the undeniable property in the standard model.

1.1. Our Contribution

Motivated by the above concerns, we construct a new SDVS scheme with undeniability and strong unforgeability, which is named the SDVS-USU scheme in this paper. The main contributions of this paper are as follows.

- The proposed scheme is the first strongly unforgeable SDVS scheme with the undeniability property in the standard model, while the existing SDVS schemes are secure in the random oracle model.
- In the SDVS-USU scheme, the signer assigns a verifier to validate the signature, and designates an arbiter to determine the actual generator of the signature. For a controversial signature, the arbiter can independently identify the real signature generator without the help of the signer or the designated verifier.
- The SDVS-USU scheme is proved to be strongly unforgeable against adaptive chosen message attacks under the bilinear Diffie–Hellman (BDH) assumption, while the privacy of the signer’s identity relies on the decisional bilinear Diffie–Hellman (DBDH) assumption. At the same time, it has the property of non-transferability.
- Compared with the existing SDVS schemes without random oracles, the SDVS-USU scheme has better performance in terms of signature length, private key size and computational cost.

1.2. Paper Organization

The rest of this paper is organized as follows. Section 2 describes the work related to SDVS. Section 3 introduces some preliminaries, such as bilinear pairings, complexity assumptions and the security definition of SDVS. Section 4 presents the SDVS-USU scheme. Section 5 demonstrates the security of the SDVS scheme. Section 6 analyzes the performance of the SDVS-USU scheme. Section 7 illustrates the application of the SDVS-USU scheme in outsourcing computation. Section 8 is the conclusions.
2. Related Work

The concept of SDVS was first introduced by Jakobsson et al. [7], and formalized by Saeednia et al. [21]. Since then, some efficient SDVS schemes were proposed [22–27], but the security of those schemes is based on the ideal random oracle. To deal with this problem, Hung et al. [17] designed a SDVS scheme in the standard model. However, its security is highly dependent on the security of the pseudo-random function. If the pseudo-random function leaks the associated index, the attacker can easily generate legitimate signatures for arbitrary messages on behalf of the signer or the designated verifier. Based on the $q$-Strong Diffie–Hellman assumption, Zhang et al. [16] constructed another SDVS scheme without random oracles. However, their SDVS scheme could not protect the privacy of the signer’s identity and did not give formal security proof. Besides, Asaar et al. [18] presented a secure SDVS scheme based on Waters’ scheme [28], but their scheme is malleable. Tian et al. [20] showed that the above three SDVS schemes [16–18] do not satisfy strong unforgeability. Later, Tian et al. [20] used the OR proof [29] and Kang et al.’s scheme [30] to design a basic signature scheme with existential unforgeability. Then, Tian et al. [20] constructed a SDVS scheme using their basic scheme and the Cramer–Shoup scheme [31]. To shorten the signature length, Tian et al. [20] proposed another SDVS scheme based on their basic signature scheme and Tian et al.’s encryption scheme [32]. Although Tian et al.’s two SDVS schemes [20] satisfy strong unforgeability, neither provides undeniability. To overcome this shortcoming, Yang et al. [12] designed an undeniable SDVS scheme using chameleon hash function [33]. However, the signer needs to store all previous signature data to identify the real generator in a signature, and the judgment process needs the help of the signer. To improve the fairness of the judgment, Hu et al. [14] designed two undeniable SDVS schemes in which the arbiter can independently identify the real signer in a disputed signature. However, Yang et al.’s scheme [12] and Hu et al.’s schemes [14] were provably secure in the random oracle model. Unfortunately, there is no strongly unforgeable SDVS scheme with the undeniable property in the standard model. Thus, in this paper, we put forward such construction for SDVS.

3. Preliminaries

3.1. Bilinear Paring

Suppose $p$ is a prime, $G_1$ and $G_2$ are two cyclic groups of order $p$, and $g$ is any generator of $G_1$. A map $e : G_1 \times G_1 \rightarrow G_2$ is called a bilinear pair if it satisfies the following conditions [18]:

- **Bilinearity:** For any $x, y \in Z_p$, $e(g^x, g^y) = e(g, g)^{xy} = e(g^y, g^x)$.
- **Non-degeneracy:** $e(g, g) \neq 1$.
- **Computability:** For any $x, y \in Z_p$, $e(g^x, g^y)$ can be calculated efficiently.

3.2. Complexity Assumptions

Given $(g, g^x, g^y, g^z) \in G_1^4$, where $x, y, z \in Z_p$ are unknown, the BDH problem is to calculate $e(g, g)^{xyz}$.

**Definition 1.** The BDH assumption is that the probability of any probabilistic polynomial-time (PPT) algorithm solving the BDH problem is negligible.

Given $(g, g^x, g^y, z) \in G_1^4$ and $Z \in G_2$, where unknown $x, y, z \in Z_p$, the DBDH problem is to determine whether $Z = e(g, g)^{xyz}$ holds.

**Definition 2.** The DBDH assumption is that there is no PPT algorithm to solve the DBDH problem with a probability of more than $\frac{1}{2}$ [18].
### 3.3. Strong Designated Verifier Signature

An SDVS scheme with undeniable property is defined as follows:

- **Setup**: On the input of a security parameter \( \lambda \in \mathbb{Z} \), this algorithm produces the public parameters \( \text{params} \).
- **KeyGen**: On the input of \( \text{params} \), this algorithm produces a public/private key pair \((pk_{S}, sk_{S})\) for a signer \( S \), \((pk_{V}, sk_{V})\) for a designated verifier \( V \) and \((pk_{A}, sk_{A})\) for an arbiter \( A \).
- **Sign**: On the input of public keys of \( S, V \) and \( A \), the signer \( S \)'s private key \( sk_{S} \) and a message \( m \), this algorithm produces a signature \( \sigma \) on \( m \).
- **Verify**: Given public keys of \( S, V \) and \( A \), this algorithm returns 1 if the designated verifier \( V \)'s private key \( sk_{V} \) can be used to verify that \( \sigma \) is a legal signature for a message \( m \); otherwise, it returns 0.
- **Sim**: On the input of public keys of \( S, V \) and \( A \), the designated verifier \( V \)'s private key \( sk_{V} \) and a message \( m \), this algorithm produces a simulated signature \( \sigma' \) that is indistinguishable from \( \sigma \).

The correctness of SDVS requires that both the original signature and the simulated signature are valid. That is, for any key pairs \((pk_{S}, sk_{S})\), \((pk_{V}, sk_{V})\) and \((pk_{A}, sk_{A})\), any message \( m \), any signature \( \sigma = \text{Sign}(pk_{S}, pk_{V}, pk_{A}, sk_{S}, sk_{V}, m) \) and any simulated signature \( \sigma' = \text{Sim}(pk_{S}, pk_{V}, pk_{A}, sk_{V}, m) \), the following two equations must hold:

\[
\text{Verify}(pk_{S}, pk_{V}, pk_{A}, sk_{V}, m, \sigma) = 1, \\
\text{Verify}(pk_{S}, pk_{V}, pk_{A}, sk_{V}, m, \sigma') = 1.
\]

A secure SDVS scheme with undeniable property should achieve the security requirements of strong unforgeability, non-transferability, privacy of the signer’s identity (PSI) and undeniability. The unforgeability requires that only the signer and the designated verifier can produce a valid signature. Formally, the strong unforgeability of an SDVS scheme is defined by the following game between a challenger \( C \) and an adversary \( F \).

- **Setup**: \( C \) executes the **Setup** algorithm to output the public parameters \( \text{params} \), and runs the **KeyGen** algorithm to generate the signer’s key pair \((pk_{S}, sk_{S})\), the designated verifier’s key pair \((pk_{V}, sk_{V})\) and the arbiter’s key pair \((pk_{A}, sk_{A})\). Then, \( C \) sends \((\text{params}, pk_{S}, pk_{V}, pk_{A})\) to \( F \).
- **Signing queries**: When \( F \) initiates a signature query for message \( m_{i} \), \( C \) runs the **Sign** \((pk_{S}, pk_{V}, pk_{A}, sk_{S}, m_{i})\) algorithm to obtain a signature \( \sigma_{i} \) on \( m_{i} \) and returns \( \sigma_{i} \) to \( F \).
- **Simulating queries**: When \( F \) asks for a simulated signature on a message \( m_{i} \), \( C \) runs the **Sim** \((pk_{S}, pk_{V}, pk_{A}, sk_{V}, m_{i})\) algorithm to obtain a signature \( \sigma'_{i} \) on \( m_{i} \) and returns \( \sigma'_{i} \) to \( F \).
- **Verifying queries**: When \( F \) submits a signature \( \sigma_{i} \) on a message \( m_{i} \), \( C \) sends the signature verification result output by the algorithm \( \text{Verify}(pk_{S}, pk_{V}, pk_{A}, sk_{V}, m_{i}, \sigma_{i}) \) to \( F \).
- **Output**: Finally, \( F \) outputs a message/signature pair \((m^{*}, \sigma^{*})\). \( F \) wins the game if

1. \( \text{Verify}(pk_{S}, pk_{V}, pk_{A}, sk_{V}, m^{*}, \sigma^{*}) = 1 \).
2. \((m^{*}, \sigma^{*})\) has not been produced by the **Simulating** query.
3. \((m^{*}, \sigma^{*})\) is not one of all tuples \((m_{i}, \sigma_{i})\) during the **Signing** queries.

**Definition 3.** If the probability of any PPT attacker \( F \) winning in the above game is negligible, then an SDVS scheme is said to be strongly unforgeable against adaptive chosen message attacks.

The non-transferability requires that no third party can tell the signature on a message was created by the signer or was simulated by the designated verifier.

**Definition 4.** An SDVS scheme is said to be non-transferable if it is not feasible for any PPT algorithm \( A_{1} \) to differentiate that a given signature is produced by the signer or the designated verifier without knowing
the signer’s private key $sk_S$, the designated verifier’s private key $sk_V$ or the arbiter’s private key $sk_A$. That is, the probability $\epsilon$ of $A_1$ distinguishing between simulated signatures and real signatures is negligible.

$$\Pr \left[ b = b' \left| \begin{array}{l}
    c_0 \leftarrow \text{Sign}(pk_S, pk_V, pk_A, sk_S, m) \\
    c_1 \leftarrow \text{Sim}(pk_S, pk_V, pk_A, sk_S, m)
  \end{array} \right. \quad b \leftarrow \{0, 1\} \\
    b' \leftarrow A_1(pk_S, pk_V, pk_A, c_b) \right] = \epsilon.$$  

In other words, the signature generated by the signer is computationally indistinguishable from the signature simulated by the designated verifier, i.e.,

$$\{\text{Sign}(pk_S, pk_V, pk_A, sk_S, m)\} \approx \{\text{Sim}(pk_S, pk_V, pk_A, sk_S, m)\}.$$  

PSI requires that no one other than the designated verifier knows the identity of the signer, but any third party is unable to identify the designated verifier and the signer. That is, if there are two signers $S_0$ and $S_1$, it is infeasible for any PPT adversary to differentiate whether the signature of a message is signed by $S_0$ or $S_1$ without knowing the designated verifier’s private key. PSI is formally defined by the following security game between a distinguisher $D$ and a challenger $B$.

- **Setup:** $B$ runs the **Setup** algorithm to produce the public parameters $params$, and runs the **KeyGen** algorithm to generate the signer $S_0$’s key pair $(pk_{S_0}, sk_{S_0})$, the signer $S_1$’s key pair $(pk_{S_1}, sk_{S_1})$, the designated verifier $V$’s key pair $(pk_V, sk_V)$ and the arbiter $A$’s key pair $(pk_A, sk_A)$. Then, $B$ sends $(params, pk_{S_0}, pk_{S_1}, pk_V, pk_A)$ to $D$.

- **Query phase 1:** $D$ adaptively initiates a series of inquiries to $B$ as follows.
  
  - **Signing queries:** When $D$ issues a signature query on a message $m_i$ and an index $d_i \in \{0, 1\}$, $B$ executes the **Sign**$(pk_{S_{d_i}}, pk_V, pk_A, sk_{S_{d_i}}, m_i)$ algorithm to obtain a signature $c_i$ on $m_i$ and returns $c_i$ to $D$.
  
  - **Simulating queries:** When $D$ issues a simulated signature query on a message $m_i$ and an index $d_i \in \{0, 1\}$, $B$ runs the **Sim**(pk_{S_{d_i}}, pk_V, pk_A, sk_V, m_i) algorithm to obtain a signature $c_i'$ on $m_i$ and returns $c_i'$ to $D$.
  
  - **Verifying queries:** After receiving a message $m_i$, a signature $c_i$ and an index $d_i \in \{0, 1\}$, $B$ responds to $D$ with the output of the algorithm **Verify**(pk_{S_{d_i}}, pk_V, pk_A, sk_V, m_i, c_i).

- **Challenge:** After receiving the challenge message $m^*$ submitted by $D$, $B$ obtains a random value $d \in \{0, 1\}$ by flipping a coin. Then, $B$ returns the signature $c^*$ generated by the algorithm **Sign**(pk_{S_{d'}}, pk_V, pk_A, sk_{S_{d'}}, m^*) to $D$.

- **Query phase 2:** $D$ continues to make queries as in Query phase 1 except that $D$ is unable to submit a signature verification query on $(m^*, c^*, d^*)$ for any $d^* \in \{0, 1\}$.

- **Output:** $D$ returns a value $d' \in \{0, 1\}$. If $d = d'$, $D$ wins the game.

**Definition 5.** An SDVS scheme is secure about PSI if there is no PPT distinguisher $D$ wins the game with a probability of more than $\frac{1}{2}$.

For a controversial signature, the undeniability requires that the arbiter can correctly identify the real identity of the generator in the signature.

**Definition 6.** An SDVS scheme is said to be undeniable if there exists a PPT arbiter, with inputting the signer’s public key $pk_S$, the designated verifier’s public key $pk_V$, the arbiter’s private key $sk_A$ and a disputed signature $\sigma$ on a message $m$, can prove whether the signer $S$ or the designated verifier $V$ generated $\sigma$ with an overwhelming probability, namely,

$$\Pr \left[ id \in \{S, V\} \leftarrow \text{Arbiter}(pk_S, pk_V, sk_A, \sigma) \right] \approx 1.$$
Here, the output $S$ indicates $\tilde{\sigma}$ is created by the signer, while the output $V$ indicates $\tilde{\sigma}$ is generated by the designated verifier.

4. The SDVS-USU Scheme

In this section, we design a strongly unforgeable SDVS scheme with undeniable property on the basis of a variant of Waters’ scheme [28]. Although a few SDVS schemes [12–14] satisfy undeniability, their security depends on ideal random oracles, which might be insecure in reality. Most of the SDVS schemes [17,18] without random oracles are malleable, so they cannot achieve strong unforgeability. To overcome these problems, the SDVS-USU scheme uses two collision-resistant hash functions to protect the integrity of the signature. This method can not only generate non-malleable signatures, but also achieve strong unforgeability and undeniability. Since we design the SDVS-USU scheme using a direct construction rather than a general conversion method, it basically maintains the performance of the Waters’ scheme [28] in terms of signature size and computational overhead. Additionally, it should be emphasized that the employed collision-resistant hash functions are not considered as random oracles in our security proof.

There are three participants in the SDVS-USU scheme: the signer $S$, the designated verifier $V$ and the arbiter $A$. In the following, we assume that all signed messages are bit strings of length $n$. To achieve this assumption, messages of arbitrary length can be converted into messages of fixed length $n$ by using a secure hash function $H : \{0,1\}^* \rightarrow \{0,1\}^n$. The SDVS-USU scheme is described as follows.

- **Setup:** Let $G_1$ and $G_2$ be two multiplicative cyclic groups of prime order $p$. $g$ is any generator of $G_1$, $e : G_1 \times G_1 \rightarrow G_2$ is a bilinear pair, and $u = (u_i)$ is a vector of length $n$, where $u_i \in G_1$.
  
  Let $H_1 : \{0,1\}^* \times G_1 \rightarrow Z_p$ and $H_2 : \{0,1\}^* \times G_1 \rightarrow Z_p$ be two collision-resistant hash functions. The public parameters are $\text{params} = (G_1, G_2, p, g, e, u_0, v, u, H_1, H_2)$.

- **KeyGen:** The signer $S$ picks two random elements $k_{S,1}, k_{S,2} \in Z_p^*$ as the private key $sk_S = (sk_{S,1}, sk_{S,2}) = (k_{S,1}, k_{S,2})$, and computes the corresponding public key $pk_S = (pk_{S,1}, pk_{S,2}) = (g^{k_{S,1}}, g^{k_{S,2}})$. Similarly, $sk_V = (sk_{V,1}, sk_{V,2}) = (k_{V,1}, k_{V,2})$ and $pk_V = (pk_{V,1}, pk_{V,2}) = (g^{k_{V,1}}, g^{k_{V,2}})$ are the designated verifier $V$’s private key and public key respectively. The arbiter $A$’s public/private key pair is $(pk_A, sk_A) = (g^{k_A}, k_A)$, where $k_A \in Z_p^*$.

- **Sign:** To generate the signature of a $n$-bit message $m = (m_1, ..., m_n) \in \{0,1\}^n$, the signer proceeds as follows.
  
  1. Select $r \in Z_p$ randomly and calculate $\sigma_2 = g^r$.
  2. Compute $w = u_0 \prod_{i=1}^{n} u_{m_i}, T = (pk_A)^{k_{S,1} k_{S,2} H_1(m_{\sigma_2})}$ and $h = H_2(m, \sigma_2, T)$.
  3. Compute $\sigma_1 = e(g^{k_{S,1} k_{S,2}}, (w v^h)^{\epsilon}, pk_{V,1})$.
  4. Output a signature $\sigma = (\sigma_1, \sigma_2, T)$ on $m$.

- **Verify:** After receiving a signature $\sigma = (\sigma_1, \sigma_2, T)$ on a $n$-bit message $m = (m_1, ..., m_n) \in \{0,1\}^n$ from the signer, the designated verifier calculates $h = H_2(m, \sigma_2, T)$ and uses its private key $sk_V = (sk_{V,1}, sk_{V,2}) = (k_{V,1}, k_{V,2})$ to verify whether
  
  $$\sigma_1 = e(pk_{S,1}, pk_{S,2})^{k_{V,1}} e(w v^h, \sigma_2)^{k_{V,2}}.$$ 

  If it holds, the designated verifier believes that $\sigma$ is legal and outputs 1; else, the designated verifier considers $\sigma$ to be illegal and outputs 0.

- **Sim:** To produce a simulated signature on a message $m = (m_1, ..., m_n) \in \{0,1\}^n$, the designated verifier performs the following:
  
  1. Select $s \in Z_p$ randomly and compute $\sigma'_2 = g^s$.
Theorem 1. If the BDH assumption holds, then the SDVS-USU scheme is strongly unforgeable against adaptive chosen message attacks in the standard model.

Proof of Theorem 1. Suppose there exists a polynomial-time adversary \( \mathcal{F} \) who breaks the strong unforgeability of the SDVS-USU scheme with non-negligible probability, where \( \mathcal{F} \) can make at most \( q_S \) signing queries, \( q_{Sim} \) simulating queries and \( q_V \) verifying queries. Then, we construct another
algorithm \( C \) who can solve the BDH problem by using the \( F \)'s forgery. Given a random BDH problem instance \((g, g^a, g^b, g^c) \in G_1^4\), the goal of \( C \) is to calculate \( e(g^a, g)^c \). \( C \) will act as \( F \)'s challenger and respond to \( F \)'s queries as follows.

- **Setup**: \( C \) simulates the algorithm \textbf{Setup} in the following way.
  1. Select \( k(0 \leq k \leq n) \) randomly, and set \( l = 4(q_5 + q_{Sim} + q_V) \) to satisfy \( l(n + 1) < p \).
  2. Select two random values \( k_1, k_2 \in Z_p \), and set the signer's public key \( pk_S = (pk_{S,1}, pk_{S,2}) = (g^{k_1}, g^{k_2}) \), the designated verifier's public key \( pk_V = (pk_{V,1}, pk_{V,2}) = (g^c, g^b) \) and the arbiiter's public key \( pk_A = g^{k_1} \). Note that \( a, b \) and \( c \) are unknown to \( C \).
  3. Select \( x_0, x_1, \ldots, x_n \in Z_p \) and \( v_0, v_1, \ldots, v_n \in Z_p \) randomly.
  4. Select a random integer \( z \in Z_p \), assign \( v = g^z \), \( u_0 = (g^b)^{p-k_1+x_0}g^{y_0} \) and \( u_j = (g^b)^{x_j}g^{y_j} \) for \( 1 \leq j \leq n \), and set a vector \( u = (u_1, \ldots, u_n) \).
  5. Pick two collision-resistant hash functions \( H_1: \{0,1\}^* \rightarrow Z_p \) and \( H_2: \{0,1\}^* \rightarrow G_1 \times G_1 \rightarrow Z_p \).
  6. Send the public parameters \( \text{params} = (G_1, G_2, p, g, e, u_0, v, u, H_1, H_2) \) and \( (pk_S, pk_V, pk_A) \) to \( F \).

For a \( n \)-bit message \( m = (m_1, \ldots, m_n) \), we define two functions

\[
F(m) = (p - lk) + x_0 + \sum_{j=1}^n x_j m_j,
\]
\[
J(m) = y_0 + \sum_{j=1}^n y_j m_j.
\]

Hence, we obtain the following equation

\[
w = u_0 \prod_{j=1}^n u_j = (g^b)^{F(m)} g^{J(m)}.
\]

- **Signing queries**: When \( F \) issues a signature query on a message \( m_i = (m_{i,1}, \ldots, m_{i,n}) \in \{0,1\}^n \), \( C \) computes the value of \( F(m_i) \). Note that if \( F(m_i) = 0 \mod p \), there exists an unique value \( 0 \leq k \leq n \) such that \( F(m_i) = 0 \mod l \). On the other hand, \( F(m_i) \neq 0 \mod l \) implies \( F(m_i) \neq 0 \mod p \). If \( F(m_i) = 0 \mod l \), \( C \) aborts. If \( F(m_i) \neq 0 \mod l \), \( C \) first searches \( m_i \) in Table \( T_r \) which is initially empty. If there is a tuple \((m_i, r_i)\) in \( T_r \), \( C \) extracts \( r_i \) from \( T_r \); otherwise, \( C \) randomly selects \( r_i \in Z_p \) and adds \((m_i, r_i)\) in \( T_r \). Then, \( C \) picks a random element \( T_i \in G_1 \), and computes \( w_i = u_0 \prod_{j=1}^n u_j m_{i,j} \),

\[
\sigma_{i,2} = (g^a)^{-\frac{r(m_i)}{r(m_j)}} g^{\sigma_{i,2}}, h_i = H_2(m_i, \sigma_{i,2}, T_i)
\]

and

\[
\sigma_{i,1} = e((g^a)^{-\frac{r(m_i)}{r(m_j)}} (w_i g^h)^{r_i}, pk_{V,1}).
\]

Finally, \( C \) returns a signature \( \sigma_i = (\sigma_{i,1}, \sigma_{i,2}, T_i) \) on \( m_i \) to \( F \).
Correctness: We show that \( \sigma_i = (\sigma_{i,1}, \sigma_{i,2}, T_i) \) is a valid signature on \( m_i \) as follows:

\[
\sigma_{i,2} = (g^d)^{r_{i,1}} \sigma_i^{T_i} = g^{r_{i,1}}
\]

\[
h_i = H_2(m_i, \sigma_{i,2}, T_i),
\]

\[
\sigma_{i,1} = e(g^a) \frac{1}{i^{H_2(m_i)}}(w_i \sigma_{i,1})^{T_i} = \sigma_i, (w_i \sigma_{i,1})^{T_i} = \sigma_i
\]

- **Simulating queries**: \( C \) responds to this kind of query in the same way as in **Signing queries**.
- **Verifying queries**: \( F \) requests a verification query on a signature \( \sigma_i = (\sigma_{i,1}, \sigma_{i,2}, T_i) \) on a message \( m_i = (m_{i,1}, \ldots, m_{i,n}) \in \{0, 1\}^n \). If \( F(m_i) = 0 \mod l \), \( C \) aborts. Otherwise, \( C \) finds \((m_i, r_i)\) in Table \( T_i \) and extracts \( r_i \) from \( T_i \). Then, \( C \) computes \( h_i = H_2(m_i, \sigma_{i,2}, T_i), F(m_i) \) and \( J(m_i) \), and checks whether

\[
\sigma_{i,1} = e(g^a) \frac{1}{i^{H_2(m_i)}}((g^b)^{F(m_i)} g^{j(m_i)} T_i r_i, \sigma_i).
\]

If this equation holds, \( C \) returns 1 to \( F \); otherwise, \( C \) returns 0 to \( F \).

- **Output**: \( F \) outputs a forged signature \( \sigma^* = (\sigma_{i,1}^*, \sigma_{i,2}^*, T^*) \) on a message \( m^* = (m_{i,1}^*, \ldots, m_{i,n}^*) \in \{0, 1\}^n \). If \( F(m^*) \neq 0 \mod p \), \( C \) aborts. Otherwise, \( C \) calculates \( w^* = \prod_{j=1}^n m_{i,j}^* \) and \( h^* = H_2(m^*, \sigma_{i,2}^*, T^*) \), and outputs \( e(g, g)^{abc} \) as follows:

\[
\frac{\sigma_{i,1}^*}{e(g^c, \sigma_{i,2}^*)} = e(g^a, g^b)^c e(w^* g^h, \sigma_i^*)
\]

\[
= e(g^a, g^b)^c e((g^b)^{F(m^*)} g^{j(m)^*} T^* \sigma_i^*)
\]

\[
= e(g^a, g^b)^c e(g^c, \sigma_i^*)^{f(m^*)+z h^*}
\]

\[
= e(g^a, g^b)^c e(g^c, \sigma_i^*)^{f(m^*)+z h^*} (since \ F(m^*) = 0 \mod p)
\]

\[
= e(g, g)^{abc}.
\]

Here, we discuss the probability of \( C \) successfully solving the BDH problem instance. If \( C \) does not abort in the above simulation, then the following conditions must hold:

- **E**: \( F(m_i) \neq 0 \mod l \) during the **Signing**, **Simulating** and **Verifying** queries.
- **E^***: \( F(m^*) = 0 \mod p \) in the forgery phase.
Our SDVS scheme is secure against the privacy of the signer’s identity under the DBDH assumption.

Theorem 3. The SDVS-USU scheme is non-transferable.

Proof of Theorem 3. Suppose there exists a PPT distinguisher that satisfies the non-transferable property. Namely, it is infeasible to distinguish the form of the original signature \( \sigma \) and the simulated signature \( \sigma' \) on \( m \) is

\[
(\sigma_1, \sigma_2, T) = (e(g^{k_{s,1}k_{s,2}}(w^h)^r, pk_{v,1}), g^r, (pk_A)^{k_{s,1}k_{s,2}H_i(m,v_2)}),
\]

and the form of the simulated signature \( \sigma' \) on \( m \) is

\[
(\sigma'_1, \sigma'_2, T') = (e(pk_{s,1}, pk_{s,2})^{k_{v,1}}e(\omega^h, \sigma'_2)^{k_{v,2}}H_i(m,v_2), g_r, (pk_A)^{k_{v,1}k_{v,2}H_i(m,v_2)}).
\]

The randomness of \( (\sigma_1, \sigma_2, T) \) determined by the random value \( r \in \mathbb{Z}_p \), and the randomness of \( (\sigma'_1, \sigma'_2, T') \) depends on the random value \( s \in \mathbb{Z}_p \). Since \( r \) and \( s \) are randomly selected from \( \mathbb{Z}_p \), the distribution of the real signature \( (\sigma_1, \sigma_2, T) \) and the simulated signature \( (\sigma'_1, \sigma'_2, T') \) is computationally indistinguishable. Namely, it is infeasible to distinguish \( \sigma \) and \( \sigma' \) without knowing the private key of the signer, the designated verifier or the arbiter. Hence, the SDVS-USU scheme satisfies the non-transferable property.

Theorem 2. The SDVS-USU scheme is non-transferable.

Proof of Theorem 2. The form of the original signature \( \sigma \) for a message \( m \) is

\[
(\sigma_1, \sigma_2, T) = (e(g^{k_{s,1}k_{s,2}}(w^h)^r, pk_{v,1}), g^r, (pk_A)^{k_{s,1}k_{s,2}H_i(m,v_2)}),
\]

and the form of the simulated signature \( \sigma' \) on \( m \) is

\[
(\sigma'_1, \sigma'_2, T') = (e(pk_{s,1}, pk_{s,2})^{k_{v,1}}e(\omega^h, \sigma'_2)^{k_{v,2}}H_i(m,v_2), g_r, (pk_A)^{k_{v,1}k_{v,2}H_i(m,v_2)}).
\]

Theorem 3. Our SDVS scheme is secure against the privacy of the signer’s identity under the DBDH assumption.

Proof of Theorem 3. Suppose there exists a PPT distinguisher \( D \) who breaks the privacy of the signer’s identity of the SDVS-USU scheme. Then, we can construct an algorithm \( B \) to solve the DBDH problem. Given a random instance \( (g, g^a, g^b, g^c, Z) \) of the DBDH problem, where unknown \( a, b, c \in \mathbb{Z}_p \) and \( Z \in G_2 \), the \( B \)'s goal is to determine if \( Z \) is equal to \( e(g, g)^{abc} \).

- Setup: \( B \) simulates the Setup algorithm by performing the following steps:
  1. Choose \( n + 2 \) random integers \( z, y_0, y_1, \ldots, y_n \in \mathbb{Z}_p \), and set \( v = g^z, u_0 = g^{y_0} \) and a vector \( \vec{u} = (u_1, \ldots, u_n) \), where \( u_i = g^{y_i} \) for \( 1 \leq j \leq n \).
  2. Select four random values \( k_1, k_2, k_3, k_4 \in \mathbb{Z}_p \), and set the signer \( S_0 \)'s public key \( pk_{S_0} = (pk_{S_0,1}, pk_{S_0,2}) = (g^a, g^b) \), the signer \( S_1 \)'s public key \( pk_{S_1} = (pk_{S_1,1}, pk_{S_1,2}) = (g^{k_1}, g^{k_2}) \),
the designated verifier’s public key $pk_V = (pk_{V,1}, pk_{V,2}) = (g^e, g^{k_1})$ and the arbiter’s public key $pk_A = g^{k_2}$. Note that $a$, $b$ and $c$ are unknown to $B$.

3. Set $sk_{S_0+V} = Z$ as the common secret key between $S_0$ and $V$, and $sk_{S_1+V} = e(g^{k_1}, g^e)^{k_2}$ as the common secret key between $S_1$ and $V$.

4. Pick two collision-resistant hash functions $H_1 : \{0,1\}^* \times G_1 \to Z_p$ and $H_2 : \{0,1\}^* \times G_1 \times G_1 \to Z_p$.

5. Send the public parameters $\text{params} = (G_1, G_2, p, g, e, u_0, v, u, H_1, H_2)$ and $(pk_S, pk_{S_1}, pk_V, pk_A)$ to $D$.

For a message $m = (m_1, \ldots, m_n) \in \{0,1\}^n$, we also define a function $L(m) = y_0 + \sum_{j=1}^{n} y_j m_j$, and thus we have $w = u_0 \prod_{j=1}^{n} u_j^{m_j} = g^{L(m)}$.

- **Query phase 1**: $B$ answers $D$’s queries as follows.
  - **Signing queries**: When $D$ issues a signature query on a message $m_i = (m_{i,1}, \ldots, m_{i,n}) \in \{0,1\}^n$ and an index $d_i \in \{0,1\}$, $B$ does the following.
    1. Select a random integer $r_i \in Z_p$, and compute $\sigma_{i,2} = g^{r_i}$.
    2. Pick a random element $T_i \in G_1$, and compute $w_i = u_0 \prod_{j=1}^{n} u_j^{m_{i,j}}$ and $h_i = H_2(m_i, \sigma_{i,2}, T_i)$.
    3. Compute $\sigma_{i,1} = sk_{S_{d_i+V}} \cdot e(w_i v^{h_i}, g^e)^{r_i}$.
    4. Return a signature $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, T_i)$ on $m_i$ to $D$.
  - **Simulating queries**: $B$ responds to this query in the same way as in Signing queries.
  - **Verifying queries**: On receiving a signature $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, T_i)$ of a message $m_i = (m_{i,1}, \ldots, m_{i,n}) \in \{0,1\}^n$ and a value $d_i \in \{0,1\}$, $B$ first calculates $w_i = u_0 \prod_{j=1}^{n} u_j^{m_{i,j}}$, $h_i = H_2(m_i, \sigma_{i,2}, T_i)$ and $L(m_i)$. Then, $B$ uses $sk_{S_{d_i+V}}$ to verify whether
    \[
    \sigma_{i,1} = sk_{S_{d_i+V}} \cdot e(\sigma_{i,2}, g^e)^{L(m_i)+zh_i}.
    \]

If this equation holds, indicating $\sigma_i$ is valid, $B$ sends 1 to $D$; otherwise, $B$ returns 0 to $D$.

**Correctness**: We show that the above signature $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, T_i)$ produced by the Signing query is correct since

\[
\sigma_{i,1} = sk_{S_{d_i+V}} \cdot e(w_i v^{h_i}, g^e)^{r_i} = sk_{S_{d_i+V}} \cdot e(g^{L(m_i)}, g^{zh_i}, g^e)^{r_i} = sk_{S_{d_i+V}} \cdot e(g^{r_i}, g^e)^{L(m_i)+zh_i} = sk_{S_{d_i+V}} \cdot e(\sigma_{i,2}, g^e)^{L(m_i)+zh_i}.
\]

- **Challenge**: When $D$ submits a challenge message $m^* = (m_{1}^*, \ldots, m_{n}^*)$, $B$ proceeds as follows:
  1. Pick $r^* \in Z_p^*$ randomly, and compute $\sigma^* = g^{r^*}$.
  2. Pick a random element $T^* \in G_1$, and compute $w^* = u_0 \prod_{j=1}^{n} u_j^{m_j^*}$ and $h^* = H_2(m^*, \sigma^*, T^*)$.
  3. Flip a fair coin to obtain a random bit $d \in \{0,1\}$.
  4. Compute $\sigma^* = sk_{S_{d+V}} \cdot e(w^* v^{h^*}, g^e)^{r^*}$.
  5. Return a signature $\sigma^* = (\sigma_1^*, \sigma_2^*, T^*)$ on $m^*$ to $D$. 


• Query phase 2: \( \mathcal{D} \) continues to issue various queries as in Query phase 1 except that \( \mathcal{D} \) cannot make a signature verification query on \((m^*, \sigma^*, d^*)\) for any \(d^* \in \{0, 1\}\).

• Output: \( \mathcal{D} \) outputs a value \(d' \in \{0, 1\}\). If \(d = d'\), indicating \(Z = e(g, g)^{abc}\), \( \mathcal{B} \) outputs 1; else, indicating \(Z\) is a random element in \(G_2\), \( \mathcal{B} \) outputs 0.

From the above simulation, we can see that \( \mathcal{B} \) does not exit in the whole simulation. Therefore, if \( \mathcal{D} \) breaks the PSI property of the SDVS-USU scheme with probability \( \epsilon \), then \( \mathcal{B} \) can solve the DBDH problem instance with probability of \( \frac{1}{2} + \epsilon \).

**Theorem 4.** The SDVS-USU scheme is undeniable.

**Proof of Theorem 4.** On receiving a disputed signature \(\sigma = (\sigma_1, \sigma_2, \tilde{T})\) on a message \(\tilde{m}\), the arbiter performs the following:

1. Obtain the signer’s public key \(pk_S = (pk_{S,1}, pk_{S,2}) = (s^{k_{S,1}}, s^{k_{S,2}})\) and the designated verifier’s public key \(pk_V = (pk_{V,1}, pk_{V,2}) = (s^{k_{V,1}}, s^{k_{V,2}})\).
2. Compute \(T_S = e(pk_{S,1}, pk_{S,2})^{k_{A}H_1(\tilde{m}\sigma_2)}\) and \(T_V = e(pk_{V,1}, pk_{V,2})^{k_{A}H_1(\tilde{m}\sigma_2)}\), where \(k_A\) is the arbiter’s private key.
3. Check \(e(\tilde{T}, g) = T_S\) or \(e(\tilde{T}, g) = T_V\). If \(e(\tilde{T}, g) = T_S\), the arbiter confirms \(\sigma\) is created by the signer. If \(e(\tilde{T}, g) = T_V\), the arbiter confirms \(\sigma\) is produced by the designated verifier.

In the proposed scheme, a signature from the signer has the form \(T = (pk_A)^{k_{S,1}k_{S,2}H_1(m\sigma_2)}\), while a signature from the designated verifier has the form \(T' = (pk_A)^{k_{V,1}k_{V,2}H_1(m\sigma_2')}\). The arbiter can independently prove the real signer of any valid signature by verifying \(e(\tilde{T}, g) = T_S\) or \(e(\tilde{T}, g) = T_V\) with probability 1. Therefore, the SDVS-USU scheme holds the undeniability property.

6. Comparison

The SDVS-USU scheme is compared with other SDVS schemes \([14,18,20]\) in terms of performance and security properties. In Tables 1 and 2, the **Size**, **Sign** and **Verify** columns represent the size of a signature, and the computational cost of signature generation and signature verification, respectively. The **SIU** column shows whether the scheme is strongly unforgeable. The **PSI** column indicates whether the scheme has the PSI property. The **Undeniability** column shows whether the scheme is undeniable. The **SM** column indicates whether the scheme is secure in the standard model. Let \(p\) and \(q\) be two primes such that \(p = 2q + 1\). Since the computational cost of some cryptographic operations such as modular multiplication, hash function or inverse is relatively small after being optimized by various technologies \([34]\), we consider only the computationally expensive bilinear pairing and exponentiation operations in Table 1. We use the symbol \(P\) to denote one paring operation. \(E_1, E_2\) and \(E_p\) denote one exponentiation operation in \(G_1, G_2, Z_p\), respectively. \(|G_1|, |G_2|, |p|\) and \(|q|\) represent the length of an element in \(G_1, G_2, Z_p\) and \(Z_q\) respectively.

| Scheme | Size | Sign | Verify |
|--------|------|------|--------|
| Scheme I in \([7]\) | 4|\(|q| + |p|\) | 5\(E_P\) | 8\(E_P\) |
| Scheme II in \([7]\) | 3|\(|q| + |p|\) | 8\(E_P\) | 9\(E_P\) |
| Asaar et al. \([11]\) | \(|G_1| + |G_2|\) | 2\(E_1 + P\) | \(E_2 + P\) |
| Scheme I in \([13]\) | \(|p| + 4|G_1|\) | 6\(E_1\) | 3\(E_1 + 2P\) |
| Scheme II in \([13]\) | \(|p| + 3|G_1|\) | 5\(E_1\) | 2\(E_1 + 2P\) |
| Our scheme | 2\(|G_1| + |G_2|\) | 4\(E_1 + P\) | \(E_1 + E_2 + P\) |
Table 2. Comparison of security properties.

| Scheme                  | SU | PSI | SM | Undeniability |
|-------------------------|----|-----|----|---------------|
| Scheme I in [7]         | Yes| No  | No | Yes           |
| Scheme II in [7]        | Yes| No  | No | Yes           |
| Asaar et al. [11]      | No | No  | Yes| No            |
| Scheme I in [13]        | Yes| Yes | Yes| No            |
| Scheme II in [13]       | Yes| Yes | Yes| No            |
| Our scheme              | Yes| Yes | Yes| Yes           |

As can be seen in Tables 1 and 2, two SDVS schemes of Hu et al. [14] outperform other schemes in both signature length and computational overhead, but their two schemes are not proven to be secure in the standard model. For the length of signature, the SDVS-USU scheme has one more element in $G_1$ than Asaar et al.’s scheme [18] but is superior to Tian et al.’s two schemes [20]. The SDVS-USU scheme is able to perform some pre-computation, such as $g^{s_1,s_2}$ in the signature generation phase and $e(pks_1,pks_2)^{vk_1}$ in the verification phase. Thus, the SDVS-USU scheme has comparable computation complexity with other schemes [18,20]. However, Asaar et al.’s scheme [18] does not have strong unforgeability and the PSI property. Moreover, none of Asaar et al.’s scheme [18] and Tian et al.’s [20] schemes holds the undeniable property. The SDVS-USU scheme has strong unforgeability and the PSI property in the standard model. Moreover, it achieves undeniability. Therefore, the SDVS-USU scheme has stronger security.

We carried out simulation experiments to evaluate the performance of the SDVS-USU scheme. The experimental environment was a laptop with Intel Core i7-6500 CPU@2.5 GHz and 8 GB memory. All simulation programs running on Microsoft Windows 10 operating system were based on PBC-0.47-VC library.

Figure 1 illustrates that the signature size of the SDVS-USU scheme, Asaar et al.’s scheme [18] and Tian et al.’s two schemes [20] is 384 bits, 256 bits, 532 bits and 404 bits, respectively. Hence, the SDVS-USU scheme has shorter signature length.

As shown in Figure 2, the length of the signer’s private key in the SDVS-USU scheme is 40 bits, which is the same as that of Asaar et al.’s scheme [18] but larger than that of Tian et al.’s two schemes [20]. Moreover, the length of the designated verifier’s private key in the SDVS-USU scheme is 40 bits, which is larger than that of Asaar et al.’s scheme [18] but smaller than that of Tian et al.’s two schemes [20].
In the signing phase, Asaar et al.’s scheme [18] requires two exponentiations and one pairing operation. The first SDVS scheme and the second SDVS scheme of Tian et al. [20] need six and five exponentiations, respectively. The SDVS-USU scheme requires four exponentiations and one pairing operation. Figure 3 shows that the computational performance of signature generation in the SDVS-USU scheme is comparable with other schemes [18,20].

In the verification phase, Asaar et al.’s scheme [18] actually requires one exponentiation and one pairing operation. The first SDVS scheme of Tian et al. [20] needs three hash functions, three exponentiations, one inverse and two pairing operations. The second SDVS scheme of Tian et al. [20] requires three hash functions, two exponentiations, one inverse and two pairing operations. The SDVS-USU scheme requires two exponentiations and one pairing operation. Figure 4 demonstrates that the computational cost of signature verification of the SDVS-USU scheme is more than that of Asaar et al.’s scheme [18] but less than Tian et al.’s two schemes [20].
7. Application in Outsourcing Computing in Cloud Computing

Cloud computing has strong computing power and storage capacity of big data. However, the cloud service provider (CSP) is not trusted by the user, and may steal the user’s private information or deceive the user. Cloud computing allows resource-constrained users to outsource expensive computations to the CSP. Hence, it is very important to ensure the integrity of the computing task and the authenticity of the remote user’s identity. Due to the limited computing ability of the user, the heavy computing task is outsourced to the CSP to complete. The CSP is able to authenticate a computing task outsourced by the user through a signature-based protocol. For the protection of private information, the user wants the designated CSP to be the only entity that can verify the legality of the signature on a computing task, and the CSP cannot reveal the signature to any third party at will. Since the ordinary digital signature has public verifiability and transferability, anyone can verify the validity of signatures by using the public key of the signer and obtain the real identity of the signer. Obviously, the ordinary digital signature scheme is not suitable for this scenario. The SDVS scheme is considered as one of the solutions to these problems, which can provide secret authentication service to the user in an outsourcing computation task. SDVS guarantees that a designated CSP can validate the user’s signature on a computing task. At the same time, it ensures that the designated CSP does not convince others that the user is involved in a computing task.

However, most of the SDVS schemes cannot identify the real signature generator when the user and the cloud service provider dispute a signature, which may cause huge economic losses to the user or the CSP. Hence, the SDVS scheme without undeniability cannot handle a controversial computing task. For example, if the user denies the submission of a computing task for some reasons, then the CSP is forced to stop it. At the same time, if the CSP forges a user’s signature on the computing task, then the user will take on the responsibility to pay for expensive computing cost. These economic losses are undesirable to the user or the CSP. The SDVS-USU scheme given in Section 3.2 is undeniable and strongly unforgeable, so it is more suitable for outsourcing computation in a cloud computing environment. The system model of outsourcing computation in cloud computing based on the SDVS-USU scheme is shown in Figure 5.
There are three entities in the system: the user, the CSP and the arbiter. The process of outsourcing calculation is as follows.

1. A user with limited computing resources uses his private key and the SDVS-USU scheme to generate a signature $\sigma_1$ for a computing task $m_1$ and sends $(m_1, \sigma_1)$ to the CSP.
2. The CSP has powerful computing power. After verifying the validity of the signature $\sigma_1$ on $m_1$ to confirm this submission, the CSP performs the computational task of $m_1$. Then, the CSP uses its private key and the SDVS-USU scheme to generate the signature $\sigma_2$ of the corresponding calculation result $m_2$, and returns $(m_2, \sigma_2)$ to the user.
3. If $\sigma_2$ is the valid signature of $m_2$, the user accepts the calculation result returned by the CSP; otherwise, the user refuses to accept $m_2$ and accuses the CSP of malicious behavior.
4. For a controversial computing task, the arbiter determines whether the user or the CSP is responsible for the economic loss of the computing task based on $(m_1, \sigma_1)$ and $(m_2, \sigma_2)$.

The SDVS-USU scheme is easily implemented as a software in cloud computing environments. For example, the signature algorithm \texttt{Sign} is installed on the user side, and the verification algorithm \texttt{Verify} is installed on the CSP side. On the one hand, the user sends the computing task and the corresponding signature to the designated CSP. On the other hand, only the designated CSP can check the integrity of the computing task and the authenticity of the user’s identity by verifying the validity of the signature, and vice versa. From the performance analysis results in Section 5, the SDVS-USU scheme has better computational performance while achieving the undeniable property. The length of the signer’s private key, the designated verifier private key and signature are 40 bits, 40 bits and 384 bits, respectively. If the message length is 900 bits, the time cost for signing and verifying is approximately 0.12 s and 0.06 s, respectively. At present, an ordinary laptop configuration is at least Intel Core i3 CPU@2.1 GHz, 4 G memory and 256 GB hard disk storage space. The CSP has more computing power, thus the SDVS-USU scheme can be practically applied to cloud computing environments.

8. Conclusions

In this paper, we construct an undeniable SDVS scheme that satisfies strong unforgeability in the standard model. The performance analysis results show that the SDVS-USU scheme has better performance in terms of private key size, signature length and computational overhead. In the SDVS-USU scheme, strong unforgeability prevents hackers from using the existing message/signature pair to create a legal signature of the same message. Non-transferability ensures that hackers cannot know the identity of the real signer in a signature. PSI further protects the privacy of the signer’s identity. Undeniability ensures that the signer and the designated verifier cannot deny messages that
they have previously sent. Therefore, our SDVS scheme can guarantee the integrity of outsourced computing tasks and authenticate the identity of users in cloud computing. In the future, we will design an instance scenario to illustrate the feasibility of implementing the SDVS-USU scheme in the real world.

**Author Contributions:** X.Y., and G.C. wrote the paper; T.L., and R.L. proved the security; and M.W. and C.W. designed the experiments.

**Funding:** This research was funded by the National Natural Science Foundation of China (No. 61662069), China Postdoctoral Science Foundation (No. 2017M610817), Science and Technology Project of Lanzhou City of China (No. 2013-4-22), and Foundation for Excellent Young Teachers by Northwest Normal University (No.NWNU-LKQN-14-7).

**Acknowledgments:** The authors would like to thank the anonymous reviewers.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**
1. Jenefa, J.; Anita, E.M. Secure vehicular communication using ID based signature scheme. *Wirel. Pers. Commun.* 2018, 98, 1383–1411. [CrossRef]
2. Mason, S. World electronic signature legislation. *Digit. Evid. Electron. Signat. Law Rev.* 2018, 15, 146–163. [CrossRef]
3. Yang, X.; Gao, G.; Zhou, Q.; Li, Y.; Wang, C. E-government data security exchange scheme based on proxy re-signature. *Comput. Eng.* 2017, 43, 183–188.
4. Baker, T.; Asim, M.; MacDermott, A.; Iqbal, F.; Kamoun, F.; Shah, B.; Alfandi, O.; Hammoudeh, M. A secure fog-based platform for SCADA-based IoT critical infrastructure. *Softw. Pract. Exp.* 2019, 2688, 1–16.
5. Tariq, N.; Asim, M.; Al-Obeidat, F.; Zubair Farooqi, M.; Baker, T.; Hammoudeh, M.; Ghafir, I. The security of big data in fog-enabled IoT applications including blockchain: A survey. *Sensors* 2019, 19, 1788. [CrossRef] [PubMed]
6. Abbas, N.; Asim, M.; Tariq, N.; Baker, T.; Abbas, S. A mechanism for securing IoT-enabled applications at the fog layer. *J. Sens. Actuator Netw.* 2019, 8, 16. [CrossRef]
7. Jakobsson, M.; Sako, K.; Impagliazzo, R. Designated verifier proofs and their applications. In Proceedings of the Eurocrypt’96, Saragossa, Spain, 12–16 May 1996; pp. 143–154.
8. Shim, K.A. On delegatability of designated verifier signature schemes. *Inf. Sci.* 2014, 281, 365–372. [CrossRef]
9. Noh, G.; Jeong, I.R. Strong designated verifier signature scheme from lattices in the standard model. *Secur. Commun. Netw.* 2016, 18, 6202–6214. [CrossRef]
10. Ghafir, I.; Saleem, J.; Hammoudeh, M.; Faour, H.; Prenosil, V.; Jaf, S.; Jabbar, S.; Baker, T. Security threats to critical infrastructure: The human factor. *J. Supercomput.* 2018, 74, 4986–5002. [CrossRef]
11. Tian, H.; Chen, X.; Zhang, F.; Wei, B.; Jiang, Z.; Liu, Y. A non-delegatable strong designated verifier signature in id-based setting for mobile environment. *Math. Comput. Model.* 2013, 58, 1289–1300. [CrossRef]
12. Yang, B.; Yu, Y.; Sun, Y. A novel construction of SDVS with secure disavowability. *Clust. Comput.* 2013, 16, 807–815. [CrossRef]
13. Hu, X.; Zhang, X.; Ma, C.; Xu, H.; Wang, J.; Tan, W. A designated verifier signature scheme with undeniable property in the random oracle. In Proceedings of the IEEE International Conference on Software Engineering and Service Science, Beijing, China, 26–28 August 2016; pp. 960–963.
14. Hu, X.; Tan, W.; Xu, H.; Wang, J.; Ma, C. Strong designated verifier signature schemes with undeniable property and their applications. *Secur. Commun. Netw.* 2017, 2017, 7921782. [CrossRef]
15. Canetti, R.; Goldreich, O.; Halevi, S. The random oracle methodology, revisited. *J. ACM* 2004, 51, 557–594. [CrossRef]
16. Zhang, H.; Ji, C. An efficient designated verifier signature scheme without random oracles. In Proceedings of the IEEE The First International Symposium on Data, Privacy, and E-Commerce, Beijing, China, 26–28 August 2016; pp. 338–340.
17. Huang, Q.; Yang, G.; Wong, D.; Susilo, W. Efficient strong designated verifier signature schemes without random oracles or delegatability. *Int. J. Inf. Secur.* 2011, 10, 373–385. [CrossRef]
18. Asaar, M.; Salmasizadeh, M. A Pairing Based Strong Designated Verifier Signature Scheme without Random Oracles. Available online: https://eprint.iacr.org/2012/061.pdf (accessed on 5 May 2019).

19. Pang, L.; Zhao, H.; Zhou, X.; Li, H. Strongly unforgeable and efficient proxy signature scheme with fast revocation secure in the standard model. Int. J. Distrib. Sens. Netw. 2016, 12, 1–12. [CrossRef]

20. Tian, H.; Jiang, Z.; Liu, Y.; Wei, B. A systematic method to design strong designated verifier signature without random oracles. Clust. Comput. 2013, 16, 817–827. [CrossRef]

21. Saeednia, S.; Kremer, S.; Markowitch, O. An efficient strong designated verifier signature scheme. In Proceedings of the 6th International Conference Information Security and Cryptology, Seoul, Korea, 27–28 November 2003; pp. 40–54.

22. Kang, B.; Boyd, C.; Dawson, E. A novel identity-based strong designated verifier signature scheme. J. Syst. Softw. 2009, 82, 270–273. [CrossRef]

23. Tian, H.; Chen, X.; Li, J. A short non-delegatable strong designated verifier signature. In Proceedings of the ACISP, Wollongong, NSW, Australia, 9–11 July 2012; pp. 261–279.

24. Sharma, N.; Sahu, R.A.; Saraswat, V.; Sharma, B.K. Adaptively secure strong designated signature. In Proceedings of the Progress in Cryptology-IndoCrypt, India, Kolkata, India, 11–14 December 2016; pp. 43–60.

25. Hu, X.; Xu, H.; Liu, Y.; Wang, J.; Tan, W.; Zhang, X. An efficient designated verifier signature scheme with pairing-free and low cost. Secur. Commun. Netw. 2016, 18, 5724–5732. [CrossRef]

26. Khan, A.U.; Ratha, B.K. A secure strong designated verifier signature scheme. Int. J. Netw. Secur. 2017, 19, 599–604.

27. Wei, J.; Yang, G.; Mu, Y. Designated verifier proxy re-signature for deniable and anonymous wireless communications. Wirel. Pers. Commun. 2017, 97, 3017–3030. [CrossRef]

28. Waters, B. Efficient identity-based encryption without random oracles. In Proceedings of the Eurocrypt’05, Aarhus, Denmark, 22–26 May 2005; pp. 114–127.

29. Cramer, R.; Damgård, I.; Schoenmakers, B. Proofs of partial knowledge and simplified design of witness hiding protocols. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 21–25 August 1994; pp. 174–187.

30. Kang, L.; Tang, X.; Lu, X.; Fan, J. A Short Signature Scheme in the Standard Model. Available online: https://pdfs.semanticscholar.org/dd25/6b488873276c7ddd7986a8ef306b045cb8d1.pdf (accessed on 5 May 2019).

31. Cramer, R.; Shoup, V. A practical public-key cryptosystem provable secure against adaptive chosen ciphertext attack. In Proceedings of the CRYPTO’98, Santa Barbara, CA, USA, 23–27 August 1998; pp. 13–25.

32. Tian, H.; Sun, X.; Wang, M. A new public key encryption scheme. J. Comput. Sci. Technol. 2007, 22, 95–102. [CrossRef]

33. Krawczyk, H.; Rabin, T. Chameleon hashing and signatures. In Proceedings of the Network and Distributed System Security Symposium, San Diego, CA, USA, 27–29 March 2000; pp. 143–154.

34. Chang, C.; Yao, S.; Yu, D. Vectorized big integer operations for cryptosystems on the intel mic architecture. In Proceedings of the 2015 IEEE 22nd International Conference on High Performance Computing, Bengaluru, India, 16–19 December 2015; pp. 194–203.

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).