A Universal Designated Verifier Signature Scheme with Non-Delegatability in the Standard Model

Parvin Rastegari[a,*], Mehdi Berenjkoub[a]

[a]*Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan 84156-83111, Iran*

Abstract

In a designated verifier signature (DVS) scheme a signer creates a signature which is only verifiable by a designated verifier. A DVS is a useful scheme for authenticating a signer without disturbing her privacy. In a universal DVS (UDVS) scheme, everyone who holds Alice's traditional signature on a message (the signature holder), is able to transform it to a DVS for a specific verifier. Non-delegatability is a critical property of a DVS scheme in applications where the responsibility of a signer is important and cannot be delegated to another entity. In this paper, we will propose a non-delegatable UDVS scheme and prove its security requirements in the standard model (without random oracles). To the best of our knowledge, our scheme is the first non-delegatable UDVS scheme and also by considering the signer herself as the signature holder, our scheme is also the first non-delegatable DVS scheme in the standard model.

Keywords: Designated Verifier Signature, Universal Designated Verifier Signature, Non-Delegatability, Random Oracle Model, Standard Model

1. Introduction

Digital signature is an important primitive in security protocols in order to provide authentication, integrity of the messages and non-repudiation [1]. In a

*Corresponding author

Email addresses: parvin.rastegari@ec.iut.ac.ir (Parvin Rastegari), brnjkb@cc.iut.ac.ir (Mehdi Berenjkoub) traditional digital signature scheme, a signer, Alice, creates a signature which is verifiable by every verifier such as Bob. Furthermore, Bob can convince every third party such as Carol that Alice has really signed a message by presenting Alice's signature on that message. This public verifiability is not a desired property in scenarios where the privacy of the signer must be preserved, such as e-votings and e-commerce applications.

In order to overcome the conflicts between the authenticity and the privacy of the signer in digital signatures, many different solutions have been proposed. In 1989, the concept of undeniable signature was presented by Chaum and Antwerpen [2]. In undeniable signatures, the verifier requires some help of the signer to verify the signature. In order to eliminate the interaction between the verifier and the signer, the notion of designated verifier signature/proof (DVS/DVP) was proposed by Jakobsson et al. [3] and independently by Chaum [4] in 1996. DVS schemes provide authentication and integrity of messages, without providing the non-repudiation property of traditional digital signatures. In a DVS scheme, a signer, Alice, can convince a designated verifier, Bob, that Alice has really signed a message while Bob cannot transfer this conviction to any third party such as Carol. As a result, the authenticity of Alice is proved to Bob and also her privacy is preserved at the same time, without any interaction between Alice and Bob.

The authors in [3] also proposed the concept of strong designated verifier signature (SDVS) scheme, in which the secret key of the verifier is required in the verification phase. The notion of universal designated verifier signature (UDVS) scheme was first proposed in 2003 by Steinfeld et al. [5]. In a UDVS scheme, everyone who holds Alice's traditional signature on a message, is able to transform it to a DVS for a specific verifier such as Bob. If the signature holder is the signer herself, the UDVS can be considered as a DVS. Many researches have been done on DVS, SDVS and UDVS schemes with different properties in different setting models, up to now [6],[7],[8],[9],[10],[11],[12],[13],[14],[15],[16],[17],[18],[19],[20],[21],[22],[23],[24],[25],[26],[27],[28],[29],[30].

In 2005, Lipmaa et al. Introduced a new security notion for DVS schemes, called non-delegatability [6]. Most of the proposed DVS, SDVS and UDVS schemes are delegatable, i.e. either the signer (Alice) or a designated verifier (Bob) is able to delegate the signing rights (for a specific designated verifier or for any designated verifiers) to a third party (Carol) without revealing her/his secret key [3],[5],[7],[17]. As mentioned in [18], delegatability is not a desired property in many scenarios. The followings are two examples which show the weaknesses of delegatable DVS schemes:

- Consider an e-voting protocol in which the voters use a DVS scheme to sign their votes for a tallier (as a designated verifier). If this DVS scheme is delegatable, a voter can delegate his signing rights to a coercer who can then sign a vote instead of the voter. As a result, this voting protocol is coercible.
- Consider an e-commerce application which uses a DVS scheme to authenticate a subscriber (the signer) to a service-provider (the designated verifier). If the subscriber is authenticated to the service-provider, she can enjoy an e-service. If the DVS scheme is delegatable, a subscriber can delegate his signing rights to a non-subscriber without revealing her secret key. As a result, a non-subsriber can obtain the e-service freely.

Although the non-delegatability has been a focus of many recent researches, it may be undesirable in some applications. However, as mentioned in [19], the non-delegatable DVS schemes can be considered as a special category which are useful in applications where the responsibility of a signer is important and cannot be delegated to another entity (such as two above mentioned scenarios). In 2014, Shim presented a discusion on delegatability of existing DVS schemes [18]. In his paper, he showed that almost all proposed DVS schemes are delegatable [3],[5],[7],[17] and there are a few number of DVS schemes that seem to be non-delegatable [20],[21]. Although there is not any reported attacks against non-delegatability on schemes in [20],[21], however their signature lengths are heavy, their signing and verification phases are inefficient and their security are proved in the random oracle model. We have found some other DVS schemes which are claimed to be non-delegatable in the literature [19],[22],[29]. All of these schemes are analyzed in the random oracle model, except the proposed scheme in [19] which is analyzed in the standard model. However, the same authors in [31] showed that the scheme they proposed in [19] is delegatable. In [18], Shim also pointed out that there have never been proposed non-delegatable UDVS schemes and he introduced that as an open problem. However, we found a UDVS scheme which is proposed by Haung et al. in 2006 in the random oracle model and is claimed to be non-delegatable [23]. Unfortunately, Haung et al.'s UDVS scheme has a considerable weakness which is everyone (not only the designated verifier) who receives the signature from the channel, can verify the signature and convince that the signer (Alice) has really signed a message for a designated verifier (Bob). Therefore, the privacy of the signer is not preserved in Haung et al.'s scheme and it is in contrast to the basic goal of a DVS scheme. In summary, we can say that there are still many open problems in this field such as:

- proposing non-delegatable UDVS schemes,
- proposing more efficient (both in the computation and the communication costs) non-delegatable (S)DVS schemes in comparison with the existing schemes,
- and proposing non-delegatable-DVS schemes in the standard model (since all existing schemes are proposed in the random oracle model and as Rogaway discussed in [32], the schemes which their security requirements are proved in the random oracle model, are not secure when the random oracles are replaced with the real world primitives (such as hash functions). Therefore, it is desirable to provide schemes without random oracles (in the standard model)).

In this paper, we propose a non-delegatable UDVS scheme and prove its security requirements in the standard model. Furthermore, since a UDVS scheme can be considered as a DVS scheme when the signature holder is the signer

herself, we will compare our scheme with a number of recently proposed non-delegatable DVS schemes [19],[20],[21],[22],[23],[24],[25],[26],[27],[28],[29]. To the best of our knowledge, our scheme is the only non-delegatable (U)DVS scheme that is analayzed in the standard model.
The rest of the paper is organized as follows. In Section 2, the formal model and the security requirements for a UDVS scheme is described. In Section 3, we propose our non-delegatable UDVS scheme. In Section 4, we analyze the security of our proposed scheme and prove its security requirements in the standard model (without random oracles). In Section 5, a comparison between our proposed scheme and some existing schemes is provided. Finally the paper is concluded in Section 6.

2. Universal Designated Verifier Signature Schemes

In this section, the formal model and the security requirements for a UDVS scheme are described.

2.1. Formal Model

A universal designated verifier signature (UDVS) scheme is included of three parties: a signer S, a signature holder SH and a designated verifier V and is defined by seven main algorithms as follows [5],[7],[23]:

- Setup: It is a probabilistic polynomial time (PPT) algorithm which takes as input a security parameter k and outputs system parameters $params$.

$$params \longleftarrow Setup(k)$$

- Signer Key Generation (SKG): It is a PPT algorithm which takes as input $params$ and outputs a private/public key pair (Sk_S, Pk_S) for the signer.

$$(Sk_S, Pk_S) \longleftarrow SKG(params)$$

- Verifier Key Generation (VKG): It is a PPT algorithm which takes as inputs $params$ and outputs a private/public key pair (Sk_V, Pk_V) for the designated verifier.

$$(Sk_V, Pk_V) \longleftarrow VKG(params)$$

- Public Signing (PS): It is a traditional digital signing which takes as input system parameters *params*, the signer's private key Sk_S, and a message m and outputs an ordinary signature (OS) σ on message m. Note that σ is publicly verifiable.

$$\sigma \longleftarrow PS(params, Sk_S, m)$$

- Public Verification (PV): It is a traditional digital signature verification which takes as input system parameters *params*, the signer's public key Pk_S, and the message/OS pair (m, σ) and outputs 1 if the OS is valid and 0 otherwise.

$$0/1 \longleftarrow PV(params, Pk_S, (m, \sigma))$$

- Designation (DS): It is a polynomial time algorithm which is performed by the signature holder SH. This algorithm takes as input system parameters *params*, the signer's public key Pk_S, the designated verifier's public key Pk_V, and the message/OS pair (m, σ) and outputs the designated verifier signature (DVS) δ.

$$\delta \longleftarrow DS(params, Pk_S, Pk_V, (m, \sigma))$$

- Designated Verification (DV): It is a deterministic polynomial time algorithm which is performed by the designated verifier V. This algorithm takes as input system parameters *params*, the signer's public key Pk_S, the designated verifier's private key Sk_V, and the message/DVS pair (m, δ) and outputs 1 if the DVS is valid and 0 otherwise.

$$0/1 \longleftarrow DV(params, Pk_S, Sk_V, (m, \delta))$$

Remark 1. Correctness must be satisfied in a UDVS scheme for both the OS and the DVS. Correctness of the OS guarantees that if σ is created by the PS

algorithm, it must be accepted in the PV algorithm, i.e.
$\Pr[1 \longleftarrow PV(params, Pk_S, (m, PS(params, Sk_S, m)))] = 1.$
Also, correctness of the DVS guarantees that if δ is created by the DS algorithm, it must be accepted in the DV algorithm, i.e.
$\Pr[1 \longleftarrow DV(params, Pk_S, Sk_V, (m, DS(params, Pk_S, Pk_V, (m, \sigma))))] = 1.$
Remark 2. Note that a UDVS scheme can be considered as a DVS scheme when the signature holder is the signer herself.

2.2. Security Requirements

Unforgeability (UF) and non-transferability (NT) are two security requirements of a UDVS scheme [5],[7],[23]. As mentioned in Section 1, non-delegatability (ND) was introduced as a new security notion for DVS schemes in 2005 [6]. In this subsection, these security requirements are described.

2.2.1. Unforgeability (UF)

In a UDVS scheme, two types of unforgeability can be considered. The first type is unforgeability of the ordinary signature (OS-Unforgeability) which is the usual existential unforgeability against chosen message attack (EUF-CMA) of the ordinary signature and guarantees that no one can forge a traditional signature σ of the signer S [1]. The second type is unforgeability of the designated verifier signature (DVS-Unforgeability) which is the existential unforgeability against chosen message attack (EUF-CMA) of the designated verifier signature and guarantees that an adversary, without having an ordinary signature σ on a message m, is not able to forge a designated verifier signature δ and convince a designated verifier of holding σ [23]. As mentioned in [23], the DVS-Unforgeability always implies the OS-Unforgeability. Therefore, considering the DVS-Unforgeability is enough. The DVS-Unforgeability is defined by the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ [5],[7],[23]:

- $\mathcal{C}$ provides $\mathcal{A}$ system parameters $params$, a public key for the signer Pk_S and a public key for the designated verifier Pk_V.
- $\mathcal{A}$ can issue queries to the following oracles, adaptively:

  - $\mathcal{O}_{PS}$: which takes as input a message m and outputs a valid OS σ on m with respect to Pk_S.
  - $\mathcal{O}_{DS}$: which takes as input a message m and outputs a valid DVS δ on m with respect to Pk_S and Pk_V.
  - $\mathcal{O}_{DV}$: which takes as input a message/DVS pair (m, δ) and outputs 1 if the DVS is valid and 0 otherwise.

- $\mathcal{A}$ outputs a DVS δ^* on message m^*.

It is said that $\mathcal{A}$ wins the above game if δ^* is a valid DVS on m^*, i.e:

$$DV(params, Pk_S, Sk_V, (m^*, \delta^*)) = 1,$$

and m^* has never been submitted as an input to the $\mathcal{O}_{PS}$ or $\mathcal{O}_{DS}$.

Definition 1. A UDVS scheme is $(t, \varepsilon, q_{PS}, q_{DS}, q_{DV})$-unforgeable, if no PPT adversary with at most q_{PS} queries from $\mathcal{O}_{PS}$, q_{DS} queries from $\mathcal{O}_{DS}$, and q_{DV} queries from $\mathcal{O}_{DV}$ can win the above game in time at most t with probability at least ε.

2.2.2. Non-Transferability (NT)

Given a valid message/DVS pair (m, δ), it must be infeasible for any PPT algorithm to distinguish whether δ is created by the signature holder SH or the designated verifier V. This property guarantees the privacy of the signer, since the designated verifier cannot convince any third party that the DVS is produced by anyone who holds the corresponding OS σ on m (including the signer herself). This concept is defined by a Transcript Simulation TS which can generate an indistinguishable DVS from that produced by SH, as follows [5],[7],[23]:

Definition 2. A UDVS scheme is non-transferable if for all PPT algorithms $\mathcal{D}$, for any security parameter k, $params \longleftarrow Setup(k)$, $(Sk_S, Pk_S) \longleftarrow SKG(params)$, $(Sk_V, Pk_V) \longleftarrow VKG(params)$, any message m, and any OS $\sigma \longleftarrow PS(params, Sk_S, m)$, the value of

$$\left| \Pr\left[\begin{pmatrix} \delta_0 \longleftarrow DS(params, Pk_S, Pk_V, (m, \sigma)) \\ \delta_1 \longleftarrow TS(params, Pk_S, Sk_V, Pk_V, m) \\ b \in_R \{0, 1\} \\ b' \longleftarrow \mathcal{D}(\delta_b, params, m, Pk_S, Sk_S, Pk_V, Sk_V) \end{pmatrix} : b = b' \right] - \frac{1}{2} \right| \quad (1)$$

is negligible.

2.2.3. Non-Delegatability (ND)

This property guarantees that if an entity can generate a valid DVS δ (without holding the corresponding OS σ), he/she must know the private key of either the signer or the designated verifier. This notion was first introduced by Lipmaa et al. [6]. Tian et al. in [24] proposed a model for this notion in which via an interaction between an extractor $\mathcal{K}$ and a signature creator $\mathcal{F}$, the extractor can extract the private key of the signer or the designated verifier. This is formally defined by the following game between $\mathcal{K}$ and $\mathcal{F}$ [19],[22]:

- $\mathcal{K}$ generates system parameters *params* and sends it to $\mathcal{F}$.
- $\mathcal{F}$ generates a public key Pk_S (or Pk_V) and sends it to $\mathcal{K}$.
- $\mathcal{K}$ generates the other public key Pk_V (or Pk_S) and sends it to $\mathcal{F}$.
- $\mathcal{K}$ sends a message m^* to $\mathcal{F}$ and asks $\mathcal{F}$ to produce a DVS δ^* on m^*.
- $\mathcal{F}$ issues queries to the $\mathcal{O}_{PS}$, $\mathcal{O}_{DS}$ and $\mathcal{O}_{DV}$ oracles (if $\mathcal{F}$ generates Pk_S) or $\mathcal{O}_{PS}$ and $\mathcal{O}_{DS}$ (if $\mathcal{F}$ generates Pk_V), adaptively. Then $\mathcal{F}$ generates a valid DVS δ^* on m^*.
- $\mathcal{K}$ generates Sk_S (or Sk_V).

Definition 3. A UDVS scheme is $(t, \varepsilon, t', \varepsilon', q_{PS}, q_{DS}, q_{DV})$-non-delegatable, if $\mathcal{K}$ can extract the private key in time t with a probability ε when $\mathcal{F}$ can generate a valid DVS δ^* on m^* in time t' with a probability ε' in the above game, where $\varepsilon \geq poly_1(\varepsilon')$ and $t \leq poly_2(t')$ for two polynomial functions $poly_1$ and $poly_2$. Note that $\mathcal{F}$ can issue at most q_{PS} queries from $\mathcal{O}_{PS}$, q_{DS} queries from $\mathcal{O}_{DS}$, and q_{DV} queries from $\mathcal{O}_{DV}$ in the above game ($q_{DV} = 0$ when $\mathcal{F}$ generates Pk_V).

3. Our non-delegatable UDVS Scheme

In 2006, authors in [7] presented the first UDVS scheme in the standard model (without random oracles). However their scheme is delegatable as discussed in [18]. As Shim pointed out in [18], presenting a non-delegatable UDVS scheme is an open problem which has not been solved yet. In this section, we modify the UDVS scheme in [7] in order to provide non-delegatability and present the first non-delegatable UDVS scheme in the standard model. Moreover, by considering the signer herself as the signature holder (see Remark 2), our scheme can be also considered as the first non-delegatable DVS scheme in the standard model. Note that all non-delegatable DVS schemes proposed up to now, are provable in the random oracle model [20],[21],[22],[23],[24],[25],[26],[27], [28],[29]. Before presenting our scheme, some preliminaries are described in the following subsection.

3.1. Preliminaries

3.1.1. Bilinear Pairings

Consider two multiplicative cyclic groups G_1 and G_2 of prime order q and let g be a generator of G_1. There exists an admissible bilinear pairing $e : G_1 \times G_1 \longrightarrow G_2$ if and only if the followings are satisfied:

1. Bilinearity: $e(g^a, g^b) = e(g, g)^{ab}$, for all $a, b \in Z_q^*$.
2. Non-degeneracy: i.e. $e(g, g) \neq 1_{G_2}$.
3. Computability: There exists an efficient algorithm for computing $e(g, g)$.

It can be referred to [33] for more details about bilinear pairings.

3.1.2. Complexity Assumptions

Some problems in bilinear pairings are considered as hard problems in complexity theory. Some of these hard problems are as follows:

- Computational Bilinear Diffie-Hellman (CBDH) Problem: On inputs $g, g^a, g^b, g^c \in G_1$, for unknown $a, b, c \in Z_q^*$, calculate $e(g, g)^{abc} \in G_2$.
- Decisional Bilinear Diffie-Hellman (DBDH) Problem: On inputs $g, g^a, g^b, g^c \in G_1$, for unknown $a, b, c \in Z_q^*$, and $Z \in G_2$, decide whether $Z = e(g, g)^{abc}$.
- Gap Bilinear Diffie-Hellman (GBDH) Problem: On inputs $g, g^a, g^b, g^c \in G_1$, for unknown $a, b, c \in Z_q^*$, calculate $e(g, g)^{abc} \in G_2$ with the help of the DBDH oracle $\mathcal{O}_{DBDH}$. The DBDH oracle $\mathcal{O}_{DBDH}$ is that given $g, g^a, g^b, g^c \in G_1$ and $Z \in G_2$, outputs 1 if $Z = e(g, g)^{abc}$ and 0 otherwise.

Definition 4. It is said that the (t, ε)-GBDH assumption holds in (G_1, G_2), if no t-time algorithm has advantage at least ε in solving the GBDH problem in (G_1, G_2).

Remark 3. The UF of our scheme is based on GBDH assumption.

3.1.3. Knowledge Extractor Assumption in the Bilinear Diffie-Hellman setting (KEA-BDH)

Let $\mathcal{K}$ be a polynomial time algorithm which on inputs (g, g^a, g^b) for $a, b \in Z_q^*$, outputs $(g^c, e(g, g)^{abc})$. The assumption is that there is another polynomial time algorithm $\mathcal{K}^*$ which takes the same inputs as $\mathcal{K}$, uses the same coins as $\mathcal{K}$, and outputs $(c, g^c, e(g, g)^{abc})$ with a probability $(1 - \rho)$, where ρ is a negligible value.

Remark 4. KEA has been proven in a generic group model [34]. The KEA-BDH which is a variant of KEA [21], is only used in the proof of non-delegatability of our scheme.

3.2. Our proposed Scheme

In this section, we propose our non-delegatable UDVS scheme. Assume that messages are bit strings of length n_m. For generality, messages can be considered of arbitrary lengths and a hash function $H_m : \{0, 1\}^* \longrightarrow \{0, 1\}^{n_m}$ can be used to convert messages to the specific length. The algorithms of our concrete scheme are as follows:

- Setup: This PPT algorithm takes a security parameter k as input and outputs system parameters $params = \{G_1, G_2, q, g, e, g_1, u', u_1, \ldots, u_{n_m}, v', v_1, \ldots, v_{n_m}\}$, in which G_1 and G_2 are two multiplicative cyclic groups with prime order q of size k, g is a generator of G_1, and $e : G_1 \times G_1 \longrightarrow G_2$ is an admissible bilinear pairing. Other parameters are random elements of G_1, i.e. $g_1, u', u_1, \ldots, u_{n_m}, v', v_1, \ldots, v_{n_m} \in_R G_1$.

- Signer Key Generation (SKG): This PPT algorithm on input $params$, picks a random $x_S \in_R Z_q^*$ as the private key of the signer Sk_S and computes the corresponding public key as $Pk_S = g^{x_S}$, then outputs $(Sk_S, Pk_S) = (x_S, g^{x_S})$.

- Verifier Key Generation (VKG): This PPT algorithm on input $params$, picks a random $x_V \in_R Z_q^*$ as the private key of the designated verifier Sk_V and computes the corresponding public key as $Pk_V = g^{x_V}$, then outputs $(Sk_V, Pk_V) = (x_V, g^{x_V})$.

- Public Signing (PS): Let m_i denotes the $i-$th bit of the message m of length n_m. The signer, with the private key Sk_S, selects a random $r \in_R Z_q^*$ and computes the ordinary signature as follows:

$$\begin{aligned}\sigma &= (\sigma_1, \sigma_2) \\ &= ((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, g^r). \end{aligned} \tag{2}$$

- Public Verification (PV): Upon receiving $(\sigma = (\sigma_1, \sigma_2))$, everyone can verify the validity of the signature by checking whether the following equation holds or not:

$$e(\sigma_1, g) = e(g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i}, Pk_S) e(u' \prod_{i=1}^{n_m} {u_i}^{m_i}, \sigma_2). \tag{3}$$

- Designation (DS): The SH who holds an OS $\sigma = (\sigma_1, \sigma_2)$, can create a DVS δ for the designated verifier V with public key Pk_V as follows:

$$\delta = (\delta_1, \delta_2) = (e(\sigma_1, Pk_V), \sigma_2). \tag{4}$$

- Designated Verification (DV): The designated verifier V can verify the validity of $\delta = (\delta_1, \delta_2)$ by his/her private key Sk_V as follows:

$$\delta_1 = e(g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i}, Pk_S)^{Sk_V} e(u' \prod_{i=1}^{n_m} {u_i}^{m_i}, \delta_2)^{Sk_V}. \tag{5}$$

Note that the correctness is satisfied for both the OS and the DVS, since according to (2) for a valid OS we have:

$$\begin{aligned} e(\sigma_1, g) &= e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, g) \\ &= e(g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i}, g^{Sk_S}) e(u' \prod_{i=1}^{n_m} {u_i}^{m_i}, g^r) \\ &= e(g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i}, Pk_S) e(u' \prod_{i=1}^{n_m} {u_i}^{m_i}, \sigma_2). \end{aligned} \tag{6}$$

Also, according to (2) and (4) for a valid DVS we have:

$$\begin{aligned} \delta_1 &= e(\sigma_1, Pk_V) \\ &= e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, g^{Sk_V}) \\ &= e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_V}, g^{Sk_S}) e((u' \prod_{i=1}^{n_m} {u_i}^{m_i})^{Sk_V}, g^r) \\ &= e(g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i}, Pk_S)^{Sk_V} e((u' \prod_{i=1}^{n_m} {u_i}^{m_i}), \delta_2)^{Sk_V}. \end{aligned} \tag{7}$$

Therefore, the correctness is satisfied for both the OS and the DVS in our UDVS scheme, since (6) and (7) imply (3) and (5), respectively.

4. Security Analysis

As mentioned in Section 2, unforgeability, non-transferability and non-delegatability are three security requirements for a UDVS scheme. In this section, we analyze these properties of our proposed scheme.

4.1. Unforgeability(UF)

In this subsection, the unforgeability of the proposed scheme is analyzed, according to Definition 1. Since Waters presented his identity based encryption scheme in the standard model in 2001 [35], many researchers have used his strategy in order to present and prove different encryption and signature schemes in the standard model [7]. We presented our UDVS scheme with some modifications of the UDVS scheme in [7] which its UF is proved with the same approach of the proof of the UF of Waters' signature. We also use the similar strategy to prove the UF of our scheme.

Theorem 1. The proposed UDVS scheme is $(t, \varepsilon, q_{PS}, q_{DS}, q_{DV})$-unforgeable, assuming that (t', ε')-GBDH assumption holds in (G_1, G_2), where

$$\varepsilon' \geq \frac{\varepsilon}{4(q_{PS} + q_{DS})(n_m + 1)},$$

$$t' \leq t + (4q_{PS} + 4q_{DS} + 4q_{DV} + 2)T_e + (1q_{DS} + 1q_{DV} + 1)T_p,$$

in which t is the required time for $\mathcal{A}$ to forge a signature, T_e and T_p denote the time for an exponentiation in G_1 and the time for a pairing in (G_1, G_2), respectively.

Proof. Suppose that there exists an adversary $\mathcal{A}$ who can $(t, \varepsilon, q_{PS}, q_{DS}, q_{DV})$ break the scheme by running the unforgeability game as Definition 1. By this assumption, we can construct an algorithm $\mathcal{B}$ which can solve a GBDH problem in time at most t' and with probability at least ε' by using $\mathcal{A}$ as a sub-routine. A random GBDH challenge $g, g^a, g^b, g^c \in G_1$ is given to $\mathcal{B}$ and $\mathcal{B}$ tries to calculate $e(g, g)^{abc} \in G_2$ with the help of the DBDH oracle $\mathcal{O}_{DBDH}$. In order to solve this problem, $\mathcal{B}$ runs $\mathcal{A}$ as a sub-routine. $\mathcal{B}$ plays the unforgeability game with $\mathcal{A}$ and simulates $\mathcal{C}$ and all oracles for $\mathcal{A}$ in this game, as follows:

- Setup: $\mathcal{B}$ sets an integer $l = 2(q_{PS} + q_{DS})$ and selects an integer $k \in \{0, 1, \ldots, n_m\}$ (n_m is the length of the message). Assume that $l(n_m+1) < q$ and as a result $0 \leq kl < q$ (Remember that q is the order of G_1 and G_2.). $\mathcal{B}$ also randomly selects $x', x_1, \ldots, x_{n_m} \in_R Z_l$, $y', y_1, \ldots, y_{n_m} \in_R Z_q$

and $z', z_1, \ldots, z_{n_m} \in_R Z_q$. These values are kept internal to $\mathcal{B}$. In order to follow the proof more easily, define three following functions:

$$J(m) = x' + \sum_{i=1}^{n_m} m_i x_i - kl,$$
$$K(m) = y' + \sum_{i=1}^{n_m} m_i y_i,$$
$$L(m) = z' + \sum_{i=1}^{n_m} m_i z_i.$$

Then $\mathcal{B}$ assigns the public key of the signer, the public key of designated verifier and other unknown system parameters as follows:

- $\mathcal{B}$ assigns the public key of the signer as $Pk_S = g^a$. (Note that g^a is one of the inputs of the GBDH problem which $\mathcal{B}$ is trying to solve it).
- $\mathcal{B}$ assigns the public key of the designated verifier as $Pk_V = g^b$.(Note that g^b is one of the inputs of the GBDH problem which $\mathcal{B}$ is trying to solve it).
- $\mathcal{B}$ sets $g_1 = g^c$. (Note that g^c is one of the inputs of the GBDH problem which $\mathcal{B}$ is trying to solve it).
- $\mathcal{B}$ assigns $u' = g_1^{x'-kl} g^{y'}$, $u_i = g_1^{x_i} g^{y_i}$ for $i = 1, 2, \ldots, n_m$, $v' = g^{z'}$ and $v_i = g^{z_i}$ for $i = 1, 2, \ldots, n_m$. By this assignment, for any message m we have:

$$u' \prod_{i=1}^{n_m} {u_i}^{m_i} = g_1^{J(m)} g^{K(m)}, \qquad v' \prod_{i=1}^{n_m} {v_i}^{m_i} = g^{L(m)}.$$

Afterwards, $\mathcal{B}$ returns Pk_S, Pk_V and $params = \{G_1, G_2, q, g, e, g_1, u', u_1, \ldots, u_{n_m}, v', v_1, \ldots, v_{n_m}\}$ to $\mathcal{A}$. From the perspective of $\mathcal{A}$, all distributions are identical to those in the real world.

- Oracle Accesses: $\mathcal{A}$ has access to the $\mathcal{O}_{PS}$, $\mathcal{O}_{DS}$ and $\mathcal{O}_{DV}$ oracles as mentioned in the unforgeability game. $\mathcal{B}$ plays the role of theses oracles, i.e. when $\mathcal{A}$ inputs its queries to these oracles, $\mathcal{B}$ will generate the corresponding outputs for $\mathcal{A}$ as follows:

– $\mathcal{O}_{PS}$. On input a message m, this oracle must output a valid ordinary signature σ on m. When $\mathcal{A}$ gives $\mathcal{O}_{PS}$ the message m as input, $\mathcal{B}$ must generate a valid $\sigma = (\sigma_1, \sigma_2)$ without knowing the private key of the signer (Note that $\mathcal{B}$ does not know a). To produce $\sigma = (\sigma_1, \sigma_2)$, $\mathcal{B}$ acts as follows:

  ∗ If $J(m) = 0 \bmod q$, $\mathcal{B}$ aborts and reports a failure.

  ∗ If $J(m) \neq 0 \bmod q$, $\mathcal{B}$ randomly selects $r \in_R Z_q^*$. Then $\mathcal{B}$ computes:

$$\begin{aligned}\sigma &= (\sigma_1, \sigma_2)\\ &= (Pk_S^{\frac{-K(m)}{J(m)}+L(m)}(u'\prod_{i=1}^{n_m} {u_i}^{m_i})^r, g^r Pk_s^{\frac{-1}{J(m)}}).\end{aligned} \tag{8}$$

  Noting (8) and Defining $\tilde{r} = r - \frac{a}{J(m)}$, we have:

$$\begin{aligned}\sigma_1 &= Pk_S^{\frac{-K(m)}{J(m)}+L(m)}(u'\prod_{i=1}^{n_m} {u_i}^{m_i})^r\\ &= (g^{-a\frac{K(m)}{J(m)}})(g^{aL(m)})(g_1^{J(m)}g^{K(m)})^r\\ &= g_1^a(g_1^{J(m)}g^{K(m)})^{\frac{-a}{J(m)}}(g^{L(m)})^a(g_1^{J(m)}g^{K(m)})^r\\ &= (g_1 g^{L(m)})^a(g_1^{J(m)}g^{K(m)})^{r-\frac{a}{J(m)}}\\ &= (g_1 v'\prod_{i=1}^{n_m} {v_i}^{m_i})^a(u'\prod_{i=1}^{n_m} {u_i}^{m_i})^{\tilde{r}}\end{aligned} \tag{9}$$

  and also:

$$\sigma_2 = g^r Pk_S^{\frac{-1}{J(m)}} = g^r g^{\frac{-a}{J(m)}} = g^{r-\frac{a}{J(m)}} = g^{\tilde{r}}. \tag{10}$$

  According to (9) and (10) and comparing them with (2), the signature $\sigma = (\sigma_1, \sigma_2)$ computed by (8), is a valid ordinary signature on m.

– $\mathcal{O}_{DS}$. On input a message m, this oracle must output a valid DVS δ on m. When $\mathcal{A}$ gives $\mathcal{O}_{DS}$ the message m as input, $\mathcal{B}$ must generate a valid $\delta = (\delta_1, \delta_2)$ without knowing the private key of the signer and the designated verifier (Note that $\mathcal{B}$ does not know a and b). To produce $\delta = (\delta_1, \delta_2)$, $\mathcal{B}$ acts as follows:

- ∗ If $J(m) = 0 \bmod q$, $\mathcal{B}$ aborts and reports a failure.
- ∗ If $J(m) \neq 0 \bmod q$, $\mathcal{B}$ computes a valid ordinary signature $\sigma = (\sigma_1, \sigma_2)$ similar to that explained in the response to $\mathcal{O}_{PS}$, and then calculates the corresponding DVS $\delta = (\delta_1, \delta_2)$, as follows:

$$\delta = (\delta_1, \delta_2) = (e(\sigma_1, Pk_V), \sigma_2).$$

– $\mathcal{O}_{DV}$. On input a message/DVS pair $(m, \delta = (\delta_1, \delta_2))$, this oracle must output 1 if δ is a valid DVS on m and 0 otherwise. When $\mathcal{A}$ gives $\mathcal{O}_{DV}$ the message/DVS pair $(m, \delta = (\delta_1, \delta_2))$ as input, $\mathcal{B}$ must verify the validity of δ without knowing the private key of the signer and the designated verifier (Note that $\mathcal{B}$ does not know a and b). To verify δ, $\mathcal{B}$ acts as follows:

- ∗ If $J(m) = 0 \bmod q$, $\mathcal{B}$ submits

$$(g, g^a, g^b, g^c, \frac{\delta_1}{e(\delta_2^{K(m)} (g^a)^{L(m)}, g^b)}), \tag{11}$$

to the DBDH oracle $\mathcal{O}_{DBDH}$ (Note that $\mathcal{B}$ is trying to solve a GBDH problem and has access to the $\mathcal{O}_{DBDH}$). Then $\mathcal{B}$ outputs 1 to $\mathcal{A}$ if the output of $\mathcal{O}_{DBDH}$ is 1 and 0 otherwise. It can be easily shown that if $(m, \delta = (\delta_1, \delta_2))$ is a valid DVS on m, then the tuple in (11) is a valid BDH tuple, as we have:

$$\begin{aligned}
&\frac{\delta_1}{e(\delta_2^{K(m)} (g^a)^{L(m)}, g^b)} \\
&= \frac{e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, Pk_V)}{e(\delta_2^{K(m)} g^{aL(m)}, g^b)} \\
&= \frac{e(g^{ac} g^{aL(m)} g^{rK(m)}, g^b)}{e(g^{rK(m)} g^{aL(m)}, g^b)} \\
&= e(g^{ac}, g^b) = e(g, g)^{abc}.
\end{aligned}$$

* If $J(m) \neq 0 \bmod q$, $\mathcal{B}$ can generate a valid DVS $\hat{\delta} = (\hat{\delta}_1, \hat{\delta}_2)$ on m as he generates the output of $\mathcal{O}_{DS}$. Afterwards, $\mathcal{B}$ submits

$$(g, Pk_V, u' \prod_{i=1}^{n_m} {u_i}^{m_i}, \frac{\delta_2}{\hat{\delta}_2}, \frac{\delta_1}{\hat{\delta}_1}), \tag{12}$$

to the DBDH oracle $\mathcal{O}_{DBDH}$. Then $\mathcal{B}$ outputs 1 to $\mathcal{A}$ if the output of $\mathcal{O}_{DBDH}$ is 1 and 0 otherwise. It can be easily shown that if $\delta = (\delta_1, \delta_2)$ is a valid DVS on m, then the tuple in (12) is a valid BDH tuple. Noting that if $\delta = (\delta_1, \delta_2)$ is a valid DVS on m, we have:

$$\begin{aligned}
&\delta_1 = e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, Pk_V),\\
&\delta_2 = g^r,
\end{aligned}$$

and similarly if $\hat{\delta} = (\hat{\delta}_1, \hat{\delta}_2)$ is a valid DVS on m, we have:

$$\begin{aligned}
&\hat{\delta}_1 = e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^{\hat{r}}, Pk_V),\\
&\hat{\delta}_2 = g^{\hat{r}}.
\end{aligned}$$

So, we have:

$$\begin{aligned}
\frac{\delta_1}{\hat{\delta}_1} &= e((u' \prod_{i=1}^{n_m} {u_i}^{m_i})^{(r-\hat{r})}, Pk_V)\\
&= e((g^{cJ(m)} g^{K(m)})^{(r-\hat{r})}, g^b),
\end{aligned}$$

so, noting that $Pk_V = g^b$, $u' \prod_{i=1}^{n_m} {u_i}^{m_i} = g^{cJ(m)} g^{K(m)}$ and $\frac{\delta_2}{\hat{\delta}_2} = g^{r-\hat{r}}$, the tuple in (12) is a valid BDH tuple.

- Forgery: Suppose that $\mathcal{A}$ forges a DVS $\delta^* = (\delta_1^*, \delta_2^*)$ on message m^*. (Remember that $\mathcal{B}$ is trying to solve a GBDH problem.) Since $\mathcal{A}$ creates $\delta^* = (\delta_1^*, \delta_2^*)$, $\mathcal{B}$ acts as follows:
  - If $J(m^*) \neq 0 \bmod q$, $\mathcal{B}$ aborts and reports a failure.
  - If $J(m^*) = 0 \bmod q$, $\mathcal{B}$ can solve the GBDH problem by obtaining $e(g, g)^{abc}$ as follows:

$$e(g,g)^{abc} = \frac{{\delta_1}^*}{e(\delta_2^{*K(m^*)} Pk_S^{L(m^*)}, Pk_V)}. \tag{13}$$

It is easy to check that (13) holds, if δ^* is a valid signature on m^*.

Time Analysis: Noting the above descriptions we can see that $\mathcal{B}$ needs a time $t' \leq t + (4q_{PS} + 4q_{DS} + 4q_{DV} + 2)T_e + (1q_{DS} + 1q_{DV} + 1)T_p$, for running the game, where t is the required time for $\mathcal{A}$ to forge a signature, T_e and T_p denote the time for an exponentiation in G_1 and the time for a pairing in (G_1, G_2), respectively.

Probability Analysis: In order to analyze the success probability of $\mathcal{B}$, we consider the events in which $\mathcal{B}$ will not abort. $\mathcal{B}$ will not abort if both the two following events happen [7]:

- E_1: $J(m) \neq 0 \bmod q$ for all queries from $\mathcal{O}_{PS}$ and $\mathcal{O}_{DS}$. Let E_{1i} denotes the event that $J(m) \neq 0 \bmod q$ in the $i-$th query from $\mathcal{O}_{PS}$ or $\mathcal{O}_{DS}$, hence $E_1 = \bigcap_{i=1}^{(q_{PS}+q_{DS})} E_{1i}$.
- E_2: $J(m^*) = 0 \bmod q$.

Noting that $J(m) = 0 \bmod q$ implies $J(m) = 0 \bmod l$ and that if $J(m) = 0 \bmod l$, there is a unique value $k \in \{0, 1, \ldots, n_m\}$ that yields $J(m) = 0 \bmod q$, we have $\Pr[J(m) = 0 \bmod q] = \frac{1}{l} \frac{1}{n_m+1}$ [7]. By defining two events E_1 and E_2 as mentioned, we have:

$$Success\ Probability\ of\ \mathcal{B} = \varepsilon' \geq \varepsilon.\Pr[E_1 \bigcap E_2], \tag{14}$$

in which ε is the least success probability of $\mathcal{A}$ to forge a DVS. Noting that E_1

and E_2 are independent events, we have:

$$\begin{aligned}\Pr[E_1 \bigcap E_2] &= \Pr[E_1]\Pr[E_2] \\ &= \Pr[\bigcap_{i=1}^{(q_{PS}+q_{DS})} E_{1i}]Pr[E_2] \\ &= (1-\Pr[\bigcup_{i=1}^{(q_{PS}+q_{DS})} \bar{E}_{1i}])(\frac{1}{l(n_m+1)}) \\ &\geq (1-\frac{(q_{PS}+q_{DS})}{l(n_m+1)})(\frac{1}{l(n_m+1)}) \\ &\geq (1-\frac{(q_{PS}+q_{DS})}{l})(\frac{1}{l(n_m+1)}) \\ &= \frac{1}{4(q_{PS}+q_{DS})(n_m+1)},\end{aligned} \tag{15}$$

where the rightmost equality is simply implied from $l = 2(q_{PS} + q_{DS})$. Noting (14) and (15), we have:

$$Success\ Probability\ of\ \mathcal{B} = \varepsilon' \geq \frac{\varepsilon}{4(q_{PS}+q_{DS})(n_m+1)},$$

as the final result. ■

4.2. Non-Transferability (NT)

In this subsection, the non-transferability of the proposed scheme is analyzed according to Definition 2.

Theorem 2. The proposed UDVS scheme is unconditionally non-transferable.

Proof. Suppose that $\delta_0 = (\delta_{0_1}, \delta_{0_2})$ is a DVS on m which is produced by the signer and $\delta_1 = (\delta_{1_1}, \delta_{1_2})$ is a designated signature on m which is produced by the transcript simulator (TS). According to Definition 2, we must prove that the value of (1) is negligible.

In order to generate δ_0, the signer, with the private key Sk_S, selects a random element $r_0 \in_R Z_q^*$ and computes $\delta_0 = (\delta_{0_1}, \delta_{0_2})$ as follows:

$$\delta_0 = (e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S}(u' \prod_{i=1}^{n_m} {u_i}^{m_i})^{r_0}, Pk_V), g^{r_0}). \tag{16}$$

In order to generate δ_1, TS picks a random $r_1 \in_R Z_q^*$ and computes $\delta_1 = (\delta_{1_1}, \delta_{1_2})$ as follows:

$$\delta_1 = (e(g_1 v' \prod_{i=1}^{n_m} v_i{}^{m_i}, Pk_S)^{Sk_V} e(u' \prod_{i=1}^{n_m} u_i{}^{m_i}, g^{r_1})^{Sk_V}, g^{r_1}). \tag{17}$$

It is easy to see that δ_0 and δ_1 have the same distributions and hence they are indistinguishable.

Suppose that a challenger $\mathcal{C}$ selects a random element $r^* \in_R Z_q^*$ an sets $\delta_2^* = g^{r^*}$, then picks a $b \in_R \{0,1\}$ by flipping a coin and sets δ_1^* as follows:

$$\delta_1^* = \begin{cases} e((g_1 v' \prod_{i=1}^{n_m} v_i{}^{m_i})^{Sk_S} (u' \prod_{i=1}^{n_m} u_i{}^{m_i})^{r^*}, Pk_V) & if\ b = 0 \\ \\ e(g_1 v' \prod_{i=1}^{n_m} v_i{}^{m_i}, Pk_S)^{Sk_V} e(u' \prod_{i=1}^{n_m} u_i{}^{m_i}, g^{r^*})^{Sk_V} & if \quad b = 1 \end{cases}. \tag{18}$$

Noting (16), (17) and (18), we have:

$$\Pr[\delta^* = \delta_0] = \Pr\begin{bmatrix} \delta_1^* = \delta_{0_1} \\ \delta_2^* = \delta_{0_2} \end{bmatrix} = \Pr[r^* = r_0] = \frac{1}{q-1},$$

$$\Pr[\delta^* = \delta_1] = \Pr\begin{bmatrix} \delta_1^* = \delta_{1_1} \\ \delta_2^* = \delta_{1_2} \end{bmatrix} = \Pr[r^* = r_1] = \frac{1}{q-1}.$$

Therefore, the distributions of δ_0 and δ_1 are identical and a distinguisher $\mathcal{D}$ cannot distinguish whether the signature is created by the signer or by TS. Hence, the signature is unconditionally non-transferable. ■

4.3. Non-Delegatability (ND)

In this subsection, we analyze the non-delegatability of our UDVS scheme according to Definition 4.

Theorem 3. Our UDVS scheme is $(t, \varepsilon, t', \varepsilon', q_{PS}, q_{DS}, q_{DV})$-non-delegatable assuming that KEA-BDH assumption holds with a probability $1 - \rho$, where

$$t \le t' + (3q_{PS} + 3q_{DS} + 2q_{DV} + 2)T_e + (1q_{DS} + 1q_{DV} + 1)T_p,$$

$$\varepsilon \geq (\frac{1}{2})^{(q_{PS}+q_{DS}+q_{DV})}\varepsilon'(1-\rho).$$

Proof. We use the KEA-BDH assumption to prove the non-delegatability of our proposed scheme. Indeed, we prove that if a PPT adversary $\mathcal{F}$ can create a valid DVS with a probability at least ε' in time at most t', then there is a polynomial time algorithm $\mathcal{K}$ which can produce $(g^{Sk_S}, e(g,g)^{abSk_S})$ (or $(g^{Sk_V}, e(g,g)^{abSk_V})$) on inputs (g, g^a, g^b) for $a, b \in Z_q^*$, with a probability at least $\varepsilon'' = (\frac{1}{2})^{(q_{PS}+q_{DS}+q_{DV})}\varepsilon'$ in time at most $t'' < t' + (3q_{PS} + 3q_{DS} + 2q_{DV} + 2)T_e + (1q_{DS} + 1q_{DV} + 1)T_p$. Therefore, according to the KEA-BDH assumption, there is another polynomial time algorithm $\mathcal{K}^*$ which takes the same inputs as $\mathcal{K}$, uses the same coins as $\mathcal{K}$, and outputs $(Sk_S, g^{Sk_S}, e(g,g)^{abSk_S})$ (or $(Sk_V, g^{Sk_V}, e(g,g)^{abSk_V})$), with a probability at least $\varepsilon''(1-\rho)$, where ρ is a negligible value.

Suppose that there is an adversary $\mathcal{F}$ which can produce a valid DVS with a probability at least ε' in time at most t'. We construct a polynomial time algorithm $\mathcal{K}$, which takes (g, g^a, g^b) as inputs and plays the no-delegatability game with $\mathcal{F}$ as follows:

- $\mathcal{K}$ sets the following assignments:
  - $\mathcal{K}$ randomly selects $z, x', y' \in_R Z_q$, and sets $g_1 = g^z$, $u' = g^{x'}$ and $v' = g^{y'}$.
  - $\mathcal{K}$ randomly selects $x_i \in_R Z_q$ and sets $u_i = g^{x_i}$, for $i = 1, 2, \ldots n_m$.
  - Suppose that the ℓ-th bit of message m^* (which $\mathcal{K}$ will send it to $\mathcal{F}$ in the fourth step of the game) is 1, i.e. $m^*_\ell = 1$. $\mathcal{K}$ sets $v_\ell = g^a$ (note that g^a is one of the inputs of $\mathcal{K}$).
  - $\mathcal{K}$ randomly selects $y_i \in_R Z_q$ and sets $v_i = g^{y_i}$, for $i = 1, 2, \ldots n_m$ and $i \neq \ell$.

  Afterwards, $\mathcal{K}$ returns $params = \{G_1, G_2, q, g, e, g_1, u', u_1, \ldots, u_{n_m}, v', v_1, \ldots, v_{n_m}\}$ to $\mathcal{F}$. From the perspective of $\mathcal{F}$, all distributions are identical to those in the real world. In order to follow the proof more easily, define two following functions:

$$J(m) = x' + \sum_{i=1}^{n_m} m_i x_i,$$
$$K(m) = y' + \sum_{i=1(i\neq\ell)}^{n_m} m_i y_i.$$

By the mentioned assignments, for any message m we have:

$$u' \prod_{i=1}^{n_m} {u_i}^{m_i} = g^{J(m)}, \qquad v' \prod_{i=1}^{n_m} {v_i}^{m_i} = g^{K(m)} g^{am_\ell}.$$

- $\mathcal{F}$ generates Pk_S and sends it to $\mathcal{K}$ (Assume that $Pk_S = g^c$ where c is unknown to $\mathcal{F}$).
- $\mathcal{K}$ sends g^b (one of its inputs) as the public key of the verifier Pk_V to $\mathcal{F}$.
- $\mathcal{K}$ sends m^* to $\mathcal{F}$ and asks $\mathcal{F}$ to produce a DVS δ^* on m^*.
- $\mathcal{F}$ issues queries to the $\mathcal{O}_{PS}$, $\mathcal{O}_{DS}$ and $\mathcal{O}_{DV}$ oracles, adaptively. $\mathcal{K}$ must answer these queries without the knowledge of the private key of the signer and the designated verifier (since b and c are unknown to $\mathcal{K}$). $\mathcal{K}$ plays the role of these oracles as follows:
  - $\mathcal{O}_{PS}$. When $\mathcal{F}$ gives $\mathcal{O}_{PS}$ the message m as input, $\mathcal{K}$ must generate a valid $\sigma = (\sigma_1, \sigma_2)$ without knowing the private key of the signer (Note that $\mathcal{K}$ does not know c). $\mathcal{K}$ acts as follows:
    * If $m_\ell = 1$, $\mathcal{K}$ aborts the simulation.
    * If $m_\ell = 0$ and as a result $v' \prod_{i=1}^{n_m} {v_i}^{m_i} = g^{K(m)}$, $\mathcal{K}$ selects a random $r \in_R Z_q^*$ and computes the ordinary signature as follows:

$$\begin{aligned}\sigma &= (\sigma_1, \sigma_2)\\ &= (pk_S^{z+K(m)} (u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, g^r),\end{aligned} \tag{19}$$

then $\mathcal{K}$ sends σ to $\mathcal{F}$. It is easy to see that (19) is a valid ordinary signature on m as:

$$\begin{aligned}\sigma_1 &= pk_S^{z+K(m)}(u'\prod_{i=1}^{n_m} u_i{}^{m_i})^r \\ &= (g^z g^{K(m)})^{Sk_S}(u'\prod_{i=1}^{n_m} u_i{}^{m_i})^r \\ &= (g_1 v'\prod_{i=1}^{n_m} v_i{}^{m_i})^{Sk_S}(u'\prod_{i=1}^{n_m} u_i{}^{m_i})^r.\end{aligned}$$

– $\mathcal{O}_{DS}$. When $\mathcal{F}$ gives $\mathcal{O}_{DS}$ the message m as input, $\mathcal{K}$ must generate a valid $\delta = (\delta_1, \delta_2)$ without knowing the private key of the signer and the designated verifier (Note that $\mathcal{K}$ does not know c and b). To answer $\mathcal{F}$, $\mathcal{K}$ acts as follows:

  * If $m_\ell = 1$, $\mathcal{K}$ aborts the simulation.
  * If $m_\ell = 0$, $\mathcal{K}$ produces an ordinary signature σ similar to that in simulating $\mathcal{O}_{PS}$, and computes the corresponding DVS, δ as follows:

$$\delta = (\delta_1, \delta_2) = (e(\sigma_1, Pk_V), \sigma_2),$$

  then sends it to $\mathcal{F}$.

– $\mathcal{O}_{DV}$. On input a message/DVS pair $(m, \delta = (\delta_1, \delta_2))$ this oracle must output 1 if δ is a valid DVS on m and 0 otherwise. When $\mathcal{F}$ gives $\mathcal{O}_{DV}$ the message/DVS pair $(m, \delta = (\delta_1, \delta_2))$ as input, $\mathcal{K}$ must verify the validity of δ without knowing the private key of the signer and the designated verifier (Note that $\mathcal{K}$ does not know c and b). To verify δ, $\mathcal{K}$ acts as follows:

  * If $m_\ell = 1$, $\mathcal{K}$ aborts the simulation.
  * If $m_\ell = 0$, $\mathcal{K}$ accepts (rejects) the signature if the following equality holds (does not hold):

$$\delta_1 = e(Pk_S^{z+K(m)}\delta_2^{J(m)}, Pk_V). \qquad (20)$$

It is easy to check the correctness of (20), since:

$$\begin{aligned}
& e(Pk_S^{z+K(m)}\delta_2^{J(m)}, Pk_V) \\
&= e(g^{Sk_S(z+K(m))}g^{rJ(m)}, Pk_V) \\
&= e((g^z g^{K(m)})^{Sk_S}(g^{J(m)})^r, Pk_V) \\
&= e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i})^{Sk_S}(u' \prod_{i=1}^{n_m} {u_i}^{m_i})^r, Pk_V) \\
&= e(\sigma_1, Pk_V) = \delta_1.
\end{aligned}$$

Finally, $\mathcal{F}$ generates a valid DVS $\delta^* = (\delta_1^*, \delta_2^*)$ on m^* and sends it to $\mathcal{K}$.

- $\mathcal{K}$ generates $e(g,g)^{abc}$ by the following equation:

$$e(g,g)^{abc} = \frac{\delta_1^*}{e(Pk_S^{z+K(m^*)}(\delta_2^*)^{J(m^*)}, Pk_V)}. \tag{21}$$

It is easy to check the correctness of (21). As mentioned before, $m_\ell^* = 1$, and as a result $v' \prod_{i=1}^{n_m} {v_i}^{m_i^*} = g^{K(m^*)}g^a$, so:

$$\begin{aligned}
& \frac{\delta_1^*}{e(Pk_S^{z+K(m^*)}(\delta_2^*)^{J(m^*)}, Pk_V)} \\
&= \frac{e((g_1 v' \prod_{i=1}^{n_m} {v_i}^{m_i^*})^{Sk_S}(u' \prod_{i=1}^{n_m} {u_i}^{m_i^*})^{r^*}, Pk_V)}{e(g^{Sk_S(z+K(m^*))}g^{r^* J(m^*)}, Pk_V)} \\
&= \frac{e((g^z g^{K(m^*)} g^a)^{Sk_S}(g^{J(m^*)})^{r^*}, Pk_V)}{e((g^z g^{K(m^*)})^{Sk_S}(g^{r^*})^{J(m^*)}, Pk_V)} \\
&= e(g^{aSk_S}, Pk_V) = e(g^{ac}, g^b) = e(g,g)^{abc}.
\end{aligned}$$

In summary, $\mathcal{K}$ outputs $(g^c, e(g,g)^{abc})$ on inputs (g, g^a, g^b). According to the KEA-BDH assumption, $\mathcal{K}$ can build another algorithm $\mathcal{K}^*$ with the same inputs and random tapes and outputs $(c, g^c, e(g,g)^{abc})$, where c is the private key of the signer.

Remark 5. The above game considers the case in which $\mathcal{F}$ produces Pk_S and $\mathcal{K}$ produces Pk_V. The other case (in which $\mathcal{F}$ produces Pk_V and $\mathcal{K}$ produces Pk_S) is similar to the above game in all steps, except in the followings:

- In step 2, $\mathcal{F}$ generates Pk_V and sends it to $\mathcal{K}$ (Assume that $Pk_V = g^c$ where c is unknown to $\mathcal{F}$).
- In step 3, $\mathcal{K}$ sends g^b (one of its inputs) as the public key of the signer Pk_S to $\mathcal{F}$.
- As a result, the last line of (21) in step 6 will be as follows:

  $$e(g^{aSk_S}, Pk_V) = e(g^{ab}, g^c) = e(g,g)^{abc},$$

  and according to the KEA-BDH assumption, $\mathcal{K}$ can build another algorithm $\mathcal{K}^*$ with the same inputs and random tapes and outputs $(c, g^c, e(g,g)^{abc})$, where c is the private key of the designated verifier.

Time Analysis: Noting the above descriptions we can see that $\mathcal{K}$ requires a time $t \leq t' + (3q_{PS} + 3q_{DS} + 2q_{DV} + 2)T_e + (1q_{DS} + 1q_{DV} + 1)T_p$, for running the game, where t' is the required time for $\mathcal{F}$ to create a DVS on m^*, T_e and T_p denote the time for an exponentiation in G_1 and the time for a pairing in (G_1, G_2), respectively.

Probability Analysis: $\mathcal{K}$ wins the non-delegatability game, if:

- $\mathcal{K}$ does not abort in the simulation,
- $\mathcal{F}$ generates a valid DVS on m^*,
- and the KEA-BDH assumption holds in (G_1, G_2) with a probability $1 - \rho$ for a negligible value of ρ.

Since these events are independent, we have:

$$Success\ Probability\ of\ \mathcal{K} = \varepsilon \geq \Pr[\overline{abort}].\varepsilon'.(1-\rho), \tag{22}$$

in which ε' is the least success probability of $\mathcal{F}$ to create a valid DVS on m^*. In order to calculate $\Pr[\overline{abort}]$, note that the probability of not aborting the

Table 1: A Comparison Between Non-Delegatable DVS Schemes

Scheme	Type	Signature-size	Sign-cost	Verify-cost	Model
[23]	UDVS	$\|1G_1\| + \|3Z_q^*\|$	$2T_{e1} + 1T_{e1}(2) + 1T_p$	$1T_{e2} + 1T_{e1}(2) + 2T_p$	ROM
[24]	IBDVS	$\|2G_1\| + \|1G_2\| + \|1Z_q^*\|$	$6T_{e1} + 2T_p$	$2T_{e1} + 3T_p$	ROM
[25]	IBDVS	$\|1G_1\| + \|4Z_q^*\|$	$1T_{e1} + 3T_{e2} + 3T_p$	$4T_{e2} + 4T_p$	ROM
[26]	SDVS	$\|4Z_{q'}^*\|$	$2T' + 1T'(2)$	$1T' + 2T'(2)$	ROM
[27]	IBDVS	$\|2G_1\| + \|2G_2\| + \|3Z_q^*\|$	$1T_{e1} + 4T_{e2} + 4T_p$	$4T_{e2} + 5T_p$	ROM
[20]	SDVS	$\|1G''\| + \|2Z_{q''}^*\|$	$2T''$	$2T''$	ROM
[29]	IBDVS	$\|4Z_{q'}^*\|$	$T' + 1T'(3)$	$2T'(3)$	ROM
[21]	IBDVS	$\|2G_1\| + \|1Z_q^*\|$	$2T_{e1} + 1T_{e2} + 2T_p$	$1T_{e2} + 3T_p$	ROM
[22]	SDVS	$\|2G_1\|$	$2T_{e1}$	$1T_{e1} + 2T_p$	ROM
ours	UDVS	$\|1G_1\| + \|1G_2\|$	$1T_{e1} + 1T_{e1}(2) + 1T_p$	$1T_{e2}(2) + 2T_p$	Standard

simulation is $\Pr[m_\ell = 0] = \frac{1}{2}$ in any queries from $\mathcal{O}_{PS}$, $\mathcal{O}_{DS}$ and $\mathcal{O}_{DV}$. Denoting the number of queries from $\mathcal{O}_{PS}$, $\mathcal{O}_{DS}$ and $\mathcal{O}_{DV}$ by q_{PS}, q_{DS} and q_{DV}, respectively we have:

$$\Pr[\overline{abort}] = (\frac{1}{2})^{(q_{PS}+q_{DS}+q_{DV})}. \qquad (23)$$

Finally noting (22) and (23) we have:

$$\varepsilon \geq (\frac{1}{2})^{(q_{PS}+q_{DS}+q_{DV})}\varepsilon'(1-\rho), \qquad (24)$$

and the proof is complete. ■

Remark 6. Suppose that $\mathcal{F}$ can create a valid DVS on m^* with a non-negligible probability (i.e. ε' is non-negligible in (24)). Then ε is non-negligible if $q_{PS} + q_{DS} + q_{DV} = order(log_2(.))$ and as a result $(\frac{1}{2})^{(q_{PS}+q_{DS}+q_{DV})}$ is non-negligible. Although we proved the non-delegatability of our UDVS scheme for logarithmic-bounded number of queries, but our scheme seems non-delegatable for polynomially-bounded number of queries, too.

5. Comparison

In this section, we will compare our scheme with other proposed non-delegatable DVS schemes. As mentioned before, Shim has discussed on non-delegatability of DVS schemes in [18]. In his paper, he has shown that almost all DVS schemes which proposed till then are delegatable except the schemes in [20],[21]. However, we have found some other DVS schemes in the literature that are claimed to be non-delegatable and are not discussed in Shim's paper [19],[22],[23],[24],[25],[26],[27],[28],[29],[30]. Among these schemes, only the schemes in [19] and [30] are analyzed in the standard model. However, the scheme in [19] is delegatable as shown in [31], and also the scheme in [30] is non-delagatable for the verification (not for signing) and it still is delegatable according to the Lipmaa et al.'s definition. We have not found any attacks against the non-delegatability of other schemes i.e. the proposed schemes in [20],[21],[22],[23],[24],[25],[26],[27],[28],[29]. However, the scheme in [28] uses a trusted third party which is not a common approach in designing digital signatures, an so we do not bring this scheme in our comparison. In Table 1, a comparison between the schemes in [20],[21],[22],[23],[24],[25],[26],[27],[29] and our scheme is provided.

The second column of Table 1 determines the types of the DVS schemes including traditional designated verifier signature (DVS), universal DVS (UDVS), strong DVS (SDVS), ID-based DVS (IBDVS) and ID-based strong DVS (IDS-DVS) schemes.

The third column of Table 1 shows the signature size of the schemes. In this column, the notation $|aG|$ shows the binary length of a elements in group G. It is clear that the smaller signature size results in the lower communication cost. Also, It is obvious that the signature size is related to the scheme's system parameters. There are three kinds of parameters as follows:

1. G_1 and G_2 are groups defined in this paper. The schemes in [21],[25],[27] are defined on these groups.
2. q' and p' are large primes such that $q'|p'-1$, and G' is a subgroup with order q' of $Z^*_{p'}$. The scheme in [26],[29] are defined on this group.

3. G'' is a group on an elliptic curve E with a large prime order q''. The scheme in [20] is defined on this group.

The fourth and fifth columns show the computation time required for the signing and verification, respectively. The notations in these columns are described as follows:

1. T_{e1} and T_{e2} show the exponentiation time in G_1 and G_2, respectively and T_p denotes the time for a pairing computation $e : G_1 \times G_1 \longrightarrow G_2$. Also, $T_{e1}(2)$ and $T_{e2}(2)$ shows the time for a double-exponentiation operation in G_1 and G_2, respectively. Note that the computation for multi-exponentiation has well-known accelerative algorithms [36].
2. T', $T'(2)$ and $T'(3)$ denote the time for an exponentiation, double-exponentiation and triple-exponentiation in G', respectively.
3. T'' denotes the time for an exponentiation in G''.

Finally, The sixth column determines whether the scheme is analyzed with or without random oracles.
The corresponding comments are considerable in Table 1:

- The scheme in [23] is the only non-delegatable UDVS scheme we have found in the literature. Our UDVS scheme has the following advantages in comparison with the UDVS scheme in [23]:

    1. We have proved the security requirements of our UDVS scheme in the standard model (without random oracle assumptions) but the scheme in [23] is proved in the random oracle model and as Rogaway discussed in [32], the schemes which their security requirements are proved in the random oracle model, are not secure when the random oracles are replaced by the real world primitives (such as hash functions).
    2. As mentioned in Section 1, a considerable weakness of the UDVS scheme in [23] is that everyone (not only the designated verifier) who receives the signature from the channel, can verify the signature

and convince that the signer (Alice) has really signed a message for a designated verifier (Bob). Our scheme does not suffer from this weakness, since the private key of the designated verifier is required in order to verify a DVS.

- Among all types of DVS schemes, only our scheme is provable in the standard model (without random oracles). As a great result, by considering the signer herself as the signature holder, our scheme is the first non-delegatable DVS scheme in the standard model.

6. Conclusion

We have proposed a non-delegatable universal designated verifier signature scheme and proved its security requirements i.e. unforgeability, non-transferability and non-delegatability in the standard model (without random oracles). Non-delegatability is a security notion for DVS schemes which guarantees that neither the signer nor the designated verifier is able to delegate the rights for creating a DVS to a third party without revealing her/his private key. This is a critical property in applications where the responsibility of a signer is important and cannot be delegated to another entity. However, in 2014, Shim studied the proposed DVS schemes and showed that almost all of them are delegatable. In this paper, we tried to solve some of the open problems in this field (as Shim mentioned them in his paper). As a result, we have presented the first non-delegatable UDVS scheme and proved its security requirements in the standard model (without random oracles). As another great result, by considering the signer herself as the signature holder, our scheme is also the first non-delegatable DVS scheme in the standard model.

References

[1] R. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM 2 (21) (1978) 120–126.

[2] D. Chaum, H. van Antwerpen, Undeniable signatures, In Advances in Cryptology, CRYPTO'89 Proceedings, Springer New York (1989) 212–216.

[3] M. Jakobsson, K. Sako, R. Impagliazzo, Designated verifier proofs and their applications, In Advances in Cryptology, EUROCRYPT'96, Springer Berlin Heidelberg (1996) 143–154.

[4] D. Chaum, Private signature and proof systems, U.S. Patent 5,493,614.

[5] R. Steinfeld, L. Bull, H. Wang, J. Pieprzyk, Universal designated-verifier signatures, In Advances in Cryptology, Asiacrypt'03, Springer Berlin Heidelberg (2003) 523–542.

[6] H. Lipmaa, G. Wang, F. Bao, Designated verifier signature schemes: attacks, new security notions and a new construction, ICALP'05, LNCS 3580 (2005) 459–471.

[7] F. Laguillaumie, B. Libert, J. J. Quisquater, Universal designated verifier signatures without random oracles or non-black box assumptions, SCN'06, LNCS, Springer-Verlag 4116 (2006) 66–77.

[8] X. Huang, W. Susilo, Y. Mu, W. Wu, Secure universal designated verifier signature without random oracles, Int. J. Inf. Secur. 7 (2008) 171–183.

[9] X. Huang, W. Susilo, Y. Mu, F. Zhang, Short (identity-based) strong designated verifier signature schemes, ISPEC'06, LNCS, Springer-Verlag 3903 (2006) 214–225.

[10] B. Kang, C. Boyd, E. Dawson, Identity-based strong designated verifier signature schemes: attacks and new construction, Comput. Electr. Eng. 35 (1) (2008) 49–53.

[11] F. Laguillaumie, D. Vergnaud, Multi-designated verifiers signatures, ICICS'04, LNCS, Springer-Verlag 3269 (2004) 495–507.

[12] J. Lee, J. Chang, Comment on saeednia et al.'s strong designated verifier signature scheme, Comput. Stand. Interfaces 31 (1) (2008) 258–260.

[13] Y. Li, H. Lipmaa, D. Pei, On delegatability of four designated verifier signatures, ICICS'05, LNCS, Springer-Verlag 3783 (2005) 61–71.

[14] S. Saeednia, S. Kramer, O. Markovitch, An efficient strong designated verifier signature scheme, ICISC'03, Springer-Verlag (2003) 40–54.

[15] W. Susilo, F. Zhang, Y. Mu, Identity-based strong designated verifier signature schemes, ACISP04, LNCS, Springer-Verlag 3108 (2004) 313–324.

[16] R. Zhang, J. Furukawa, H. Imai, Short signature and universal designated verifier signature without random oracles, ACNS, LNCS'05, Springer-Verlag 3531 (2005) 483–498.

[17] J. Zhang, J. Mao, A novel id-based designated verifier signature scheme, Inf. Sci. 178 (3) (2008) 766–773.

[18] K. A. Shim, On delegatability of designated verifier signature schemes, Information Sciences 281 (2014) 365–372.

[19] H. Tian, Z. Jiang, Y. Liu, B. Wei, A non-delegatable strong designated verifier signature without random oracles, In Proceedings of the 4th International Conference on Intelligent Networking and Collaborative Systems (2012) 237–244.

[20] H. Tian, X. Chen, Z. Jiang, Y. Du, Non-delegatable strong designated verifier signature on elliptic curves, In ICISC'11, LNCS 7259 (2012) 219–234.

[21] H. Tian, X. Chen, F. Zhang, B. Wei, Z. Jiang, Y. Liu, A non-delegatable strong designated verifier signature in id-based setting for mobile environment, Math. Comput. Modell. 58 (2013) 1289–1300.

[22] H. Tian, L. Jin, A short non-delegatable strong designated verifier signature, Frontiers of Computer Science 8 (3) (2014) 490–502.

[23] Y. M. X. Huang, W. Susilo, W. Wu, Universal designated verifier signature without delegatability, Lecture Notes in Computer Science 4307 (2006) 479–498.

[24] B. Wang, A non-delegatable identity-based strong designated verifier signature scheme, IACR Cryptology ePrint Archive 507.

[25] Q. Huang, W. Susilo, D. S. Wong, Non-delegatable identity-based designated verifier signature, IACR Cryptology ePrint Archive 367.

[26] Q. Huang, G. Yang, D. S. Wong, W. Susilo, Efficient strong designated verifier signature schemes without random oracle or with non-delegatability, International Journal of Information Security 10 (6) (2011) 373–385.

[27] Q. Huang, G. Yang, D. S. Wong, W. Susilo, Identity-based strong designated verifier signature revisited, Journal of Systems and Software, 84(1): pp. 120129, 2011. 84 (1) (2011) 120–129.

[28] M. R. Asaar, A. Vardasbi, M. Salmasizadeh, Non-delegatable strong designated verifier signature using a trusted third party without pairings, In Proceedings of the Eleventh Australasian Information Security, Australian Computer Society, Inc. 138 (2013) 13–24.

[29] M. R. Asaar, M. Salmasizadeh, A non-delegatable identity-based designated verifier signature scheme without bilinear pairings, IACR Cryptology ePrint Archive 332.

[30] M. R. Asaar, M. Salmasizadeh, A pairing based strong designated verifier signature scheme without random oracles, Cryptology ePrint Archive 061.

[31] H. Tian, Z. Jiang, Y. Liu, B. Wei, A systematic method to design strong designated verifier signature without random oracles, Cluster Computing, pp. 111, 2011. (2011) 1–11.

[32] M. Bellare, P. Rogaway, Random oracles are practical: A paradigm for designing efficient protocols, In Proceedings of the 1st ACM conference on Computer and communications security, ACM (1993) 62–73.

[33] D. Boneh, M. Franklin, Identity-based encryption from the weil pairing, In Advances in Cryptology, CRYPTO'01, Springer Berlin Heidelberg (2001) 213–229.

[34] A. W. Dent, S. D. Galbraith, Hidden pairings and trapdoor ddh groups, Lecture Notes in Computer Science 4076 (2006) 436–451.

[35] B. Waters, Efficient identity-based encryption without random oracles, In Advances in Cryptology, EUROCRYPT'05, Springer Berlin Heidelberg (2005) 114–127.

[36] B. Moller, Algorithms for multi-exponentiation, Lecture Notes in Computer Science 2259 (2001) 165–180.