Non-Delegatable Strong Designated Verifier Signature Using a Trusted Third Party without Pairings

Maryam Rajabzadeh Asaar[1], Ali Vardasbi[2], Mahmoud Salmasizadeh[2] *
[1] Department of Electrical Engineering, [2] Electronics Research Institute, Sharif University of Technology, Tehran, Iran.
asaar@ee.sharif.ir, vardasbi@alum.sharif.edu, salmasi@sharif.edu

Abstract Strong designated verifier signature (SDVS) is characterized by two properties; namely the non-transferability and the privacy of the signer's identity (PSI). Non-transferability prevents anyone else other than the designated verifier to verify the signature, while PSI prevents a third party to distinguish between two different signers. In this paper, we propose a non-delegatable SDVS which uses a trusted third party for the key generation. Our signature scheme does not use bilinear pairings which makes it suitable for the resource constraint applications. Using one-way homomorphic functions, our scheme is presented at an abstract level, the unification of which was noticed by Maurer in the context of zero knowledge proofs of knowledge in Africacrypt 2009. The security of the proposed scheme is proved in the random oracle model, provided that the homomorphism one-wayness and the gap Diffie-Hellman assumptions hold. When a Schnorr-like homomorphism is used to construct our scheme, six exponentiations are needed in the signing step and seven for the verification step. This means a meaningful gap between the performance of our scheme and that of its predecessors which use pairings in their signing and/or verification steps.

Keywords strong designated verifier signature, trusted third party, non-delegatability, random oracle model

1 Introduction

The notion of *designated verifier proofs* (DVP) which was introduced by Jakobson, Sako, and Impagliazzo (1996), allows the prover to designate a verifier as the only one by whom the proof can be verified. In other words, the conviction of the designated verifier is *non-transferable* to anyone else. As for many signature schemes which are non-interactive versions of some zero knowledge proofs, the *designated verifier signatures* (DVS) are the non-interactive versions of DVP. Applications of DVS include but are not limited to undeniable signature (Huang, Mu, Susilo, and Wu 2007) and deniable authentication (Wang, and Song 2009).

In addition to the non-transferability which is the property of DVS, one can think of another property which states that no third party can distinguish between different signers by looking at the signature. The concept of such a DVS for which a third party cannot tell if a designated signature for Bob is from Alice or from some other signer, was first noticed by Jakobson, Sako, and Impagliazzo (1996) and the scheme is called *strong designated verifier signature* (SDVS). Later on, SDVS was formalized by Laguillaumie, and Vergnaud (2004) where the property of *privacy of signer's identity* (PSI) was defined.

A DVS is either delegatable or non-delegatable. In a delegatable DVS the signer can delegate her signing capability to a third party without revealing her secret key. Non-delegatable DVS, on the other hand, prevents the signer from such a delegation. Formally, as was introduced by Lipmaa, Wang, and Bao (2005), the non-delegatability of a DVS necessitates the knowledge of the secret key of either Alice or Bob in order to generate a valid signature on behalf of Alice for the designated verifier Bob.

Huang, Yang, Wong, and Susilo (2011a) propose two different SDVS schemes: a delegatable SDVS, provably secure in the standard model; and a non-delegatable SDVS provably secure in the random oracle model.

Shamir introduced the concept of *identity-based signature* (IBS) by suggesting to use the identity of the signer as the verification key (Shamir 1985). There are several IBS schemes based on factoring and RSA (e.g. Shamir 1985, Fiat, and Shamir 1987, Guillou, and Quisquater 1990, and Okamoto 1993) and many IBS schemes based on pairings (e.g. Sakai, Ohgishi, and Kasahara 2000, Hess 2003, Cha, and Cheon 2002). A framework for deriving security proofs for IBS and identification schemes has been provided by Bellare, Namprempe, and Neven 2004. Galindo and Garcia (2009) propose a provable secure IBS by using sequentially delegating Schnorr signatures (Schnorr 1991) and they claim their scheme is among the most efficient provably secure IBS schemes to that date.

There are some studies which propose DVS schemes in the identity based setting with the title of IB(S)DVS. However, most of these IB(S)DVS schemes use pairings in their signature generation and verification (Huang, Susilo, Mu, and Zhang 2008, Huang, Susilo, and Wong 2009, Huang, Yang, Wong, and Susilo 2011b, Kang, Boyd, and Dawson 2009) while the others use pairing only for their verification (Cao, and Cao 2009, Zhang,

* This research is partially supported by the Office of Vice-President for the Science and Technology and Iran Telecommunication Research Center through grant no. 15712.

and Mao 2008). As the computation cost of pairings is approximately 20 times higher than that of exponentiation at the same security level (ECRYPT 2006), the significance of our scheme whose construction does not require pairings is evident. In case of the signature size, IBSDVS schemes which do not support non-delegatability (Huang, Susilo, Mu, and Zhang 2008, Kang, Boyd, and Dawson 2009, Zhang, and Mao 2008) have a smaller size (one, two and three elements, respectively) compared to the ones which are non-delegatable. The scheme proposed by Huang, Susilo, and Wong (2009) which is an IBDVS with non-delegatability has five output elements (the same as ours), but uses three pairings for signing and four pairings for verification and does not support PSI. Huang, Yang, Wong, and Susilo (2011b) propose an IBSDVS scheme with non-delegatability which has seven output elements (more than ours), uses three pairings for signing and five pairings for verification.

Our scheme shares similarity with the IBDVS scheme proposed by Rajabzadeh Asaar, Salmasizadeh (2012), but their scheme is not a strong DVS; i.e. does not support the PSI property. There are also some identity based schemes in the multi-DVS setting such as the schemes in Chow (2008) and Chow (2006). For instance, Chow (2008) proposes two generic constructions of MDVS, one of which enables many ID-based ring signature schemes to support anonymous subset.

In this paper, we propose a non-delegatable SDVS using a trusted third party without pairings and prove its security in the random oracle model. Our proposed scheme can be instantiated for any one-way group homomorphic where GDH assumption is believed to be true. It has five output elements and it uses six exponentiations for signing and seven exponentiations for verification. As a price for not using the costly bilinear pairings, our scheme is not a pure identity based scheme. However, it shares some features with the IB schemes and uses a trusted third party for delivering the keys to the signer and verifier.

The next section contains some preliminaries, definitions, notations and assumptions which are used through the rest of the paper. Section 3 is the main part of the paper which is devoted to an abstract scheme for a light-weight IBS, a modified version of a non-delegatable SDVS and our proposed scheme. Furthermore, the security analysis of scheme is presented in this section. Finally, the concluding remarks and future work are appeared in section 4.

2 Preliminaries and Notations

This section contains the notations, definitions and assumptions which are used in the rest of the paper. Furthermore, the abstract model of (Maurer 2009), which is exploited in the construction of our scheme, is reviewed in this section.

2.1 An Abstract Model

As was noted by Maurer (2009), many protocols for the zero knowledge proof of knowledge (ZKPK) can be unified as proofs of knowledge of a preimage of a group homomorphism. Since a considerable number of signatures in the literature use a (non-interactive) ZKPK, the abstract model of (Maurer 2009) can be exploited in the context of signature schemes as well. Here we briefly explain this abstract model.

Consider two groups $(\mathcal{H},\oplus)$ and $(\mathcal{G},\otimes)$ and a homomorphism $f:\mathcal{H}\to\mathcal{G}$ as follows:

$$\forall x,y\in\mathcal{H}: f(x\oplus y)=f(x)\otimes f(y) \tag{1}$$

As in (Maurer 2009), we write $[x]$ instead of $f(x)$ for simplicity and we will consider the case where f is (believed to be) a one-way function, such that it is infeasible to compute $x\xleftarrow{\$}\mathcal{H}$ from $[x]$. Furthermore, we use the following notations:

$$\forall x\in\mathcal{H}: (k\cdot x)\triangleq x\oplus\cdots\oplus x\ \ (k\text{ times}) \tag{2}$$

$$\forall X\in\mathcal{G}: \quad X^c\triangleq X\otimes\cdots\otimes X\ \ (c\text{ times}) \tag{3}$$

Two of the most popular instantiations of this model are as follows:

1) For a prime q, assume $\mathcal{H}=\mathbb{Z}_q$ is an additive group and $\mathcal{G}$ is a multiplicative group of order q and generator g. In this setting, a homomorphism can be defined as follows:

$$\forall x\in\mathbb{Z}_q: [x]=g^x \tag{4}$$

In this case, the homomorphism one-wayness assumption is equivalent to the discrete logarithm assumption. This homomorphism is exploited in the Schnorr protocol (Schnorr 1991) and in the ElGamal cryptosystem (ElGamal 1984). In the rest of this paper, we refer to this homomorphism by Schnorr-like homomorphism.

2) For large primes p and q, assume $n=pq$ and $(\mathcal{H},\oplus)=(\mathbb{Z}_n^*,\cdot)=(\mathcal{G},\otimes)$. For a given prime exponent e (with $\gcd(e,\varphi(n))=1$), a homomorphism can be defined as follows:

$$\forall x\in\mathbb{Z}_n^*: [x]=x^e \tag{5}$$

In this case, the homomorphism one-wayness assumption is equivalent to compute e-th roots modulo n without knowing the factor of n, which means breaking the RSA cryptosystem. This homomorphism is exploited in the Guillou-Quisquater (GQ) protocol (Guillou, and Quisquater 1988). In the rest of this paper, we refer to this homomorphism by GQ-like homomorphism.

2.2 Notations

Throughout this paper, S is the signer and V is the verifier. The secret keys of S and V are represented by x_s and x_v respectively. It is assumed that a common key is shared between S and V through a Diffie-Hellman-like protocol (Diffie, and Hellman 1976). It means that the common key should be feasibly computable from either one of these two pairs: $([x_s],x_v)$ or $([x_v],x_s)$, but not

from $([x_v], [x_s])$. In what follows, the common key shared between S and V with the above property is shown by:

$$\mathcal{DH}\{[x_s], x_v\} = \mathcal{DH}\{[x_v], x_s\} \quad (6)$$

In the case of a Schnorr-like homomorphism, the description of $\mathcal{DH}\{[x_s], x_v\}$ is straight forward:

$$\mathcal{DH}\{[x_s], x_v\} = g^{x_s x_v} = [x_s x_v] = \mathcal{DH}\{[x_v], x_s\} \quad (7)$$

However, in the case of a GQ-like homomorphism, one cannot simply consider $\mathcal{DH}\{[x_s], x_v\} = [x_s x_v] = (x_s x_v)^e$, since $[x_s x_v] = [x_s] \cdot [x_v]$. In this case, it requires some more thought to propose a suitable function for $\mathcal{DH}\{[x_s], x_v\}$. A trivial solution is proposed in the appendix A.

We use the notations $\mathcal{E}nc_{[x]}\{w, r\}$ and $\mathcal{D}ec_x\{W, [r]\}$ for public key encryption and decryption with the public/private key pair $(x, [x])$ and the randomizer r. These functions are defined as follows[1]:

$$\mathcal{E}nc_{[x]}\{w, r\} \triangleq w \otimes \mathcal{DH}\{[x], r\} \quad (8)$$

$$\mathcal{D}ec_x\{W, [r]\} \triangleq W \otimes (\mathcal{DH}\{[r], x\})^{-1} \quad (9)$$

2.3 Definitions

In what follows, a formal definition for the unforgeability of a SDVS scheme, non-transferability, non-delegatability and PSI is given. Further discussions and remarks on these definitions can be found in (Huang, Yang, Wong, and Susilo 2011a).

Definition 1. (Unforgeability) An IBSDVS scheme is $(t, q_{Ext}, q_{Sign}, q_{Sim}, \epsilon)$-unforgeable if no adversary $\mathcal{A}$ which runs in time at most t; issues at most q_{Ext} queries to $\mathcal{O}_{Ext}$ issues at most q_{Sign} queries to $\mathcal{O}_{sign}$; and issues at most q_{Sim} queries to $\mathcal{O}_{sim}$ can win the following game with probability at least ϵ.

This is the game which is considered to be played between the challenger $\mathcal{B}$ and a probabilistic polynomial time (PPT) adversary $\mathcal{A}$:

i. $\mathcal{B}$ runs the **Gen** algorithm to generate a master key pair (x_m, pk_m), and gives pk_m to $\mathcal{A}$.
ii. $\mathcal{A}$ issues queries to the following oracles:
   - $\mathcal{O}_{Ext}$: This oracle returns the user's secret key $([a], x) \leftarrow$ **Ext**(x_m, pk_m, id) on a given id.
   - $\mathcal{O}_{sign}$: Given a query of the form of (x_s, id_s, id_v, M), this oracle signs the message M as $\sigma \leftarrow$ **Sign**$(x_s, id_s, id_v, pk_m, M)$, and returns it to $\mathcal{A}$.
   - $\mathcal{O}_{sim}$: Given a query of the form of (x_s, id_s, id_v, M) this oracle simulates the $\mathcal{O}_{sign}$ in order to output the corresponding signature.
iii. $\mathcal{A}$ outputs a forgery $(M^*, id_s^*, id_v^*, \sigma^*)$ and wins the game if the three following conditions hold
   - **Ver**$(\sigma^*, id_s^*, id_v^*, M^*) = 1$,
   - $\mathcal{A}$ did not query $\mathcal{O}_{Ext}$ on input id_s^* or id_v^*, and
   - $\mathcal{A}$ did not query $\mathcal{O}_{sign}$ and $\mathcal{O}_{sim}$ on input $(M^*, id_s^*, id_v^*, \sigma^*)$.

Definition 2. (Non-transferability). An IBSDVS is non-transferable if there exists a PPT simulation algorithm **Sim** on x_v, id_s, id_v, and a message M outputs a simulated signature which is indistinguishable from the real signatures generated by the signer on the same message. For any PPT distinguisher $\mathcal{D}$, any (id_s, x_s), (id_v, x_v), and any message M, it holds that

$$\left|\Pr[b' \leftarrow \mathcal{D}(id_s, id_v, pk_m, M, \sigma_b) : b' = b] - \frac{1}{2}\right| < \epsilon(k)$$

where $\sigma_0 \leftarrow$ **Sign**$(x_s, id_s, id_v, pk_m, M)$, $\sigma_1 \leftarrow$ **Sim**$(x_v, id_s, id_v, pk_m, M)$, $b \xleftarrow{\$} \{0,1\}$ and $\epsilon(k)$ is a negligible function in the security parameter k, and the probability is taken over the randomness used in **Sign** and **Sim**, and the random coins consumed by $\mathcal{D}$. If the probability is equal to $1/2$, the IBSDVS scheme is perfectly non- transferable.

Definition 3. (Non-delegatability). It is assumed that $\kappa \in [0, 1]$ be the knowledge error. An IBSDVS scheme is (t, κ)-non-delegatable if there is a black box knowledge extractor which produces either the secret key of the signer or the secret key of the designated verifier with oracle access to the forger $\mathcal{F}$. If $\mathcal{F}$ generates a valid signature with probability ϵ on a message M for every $(x_m, pk_m) \leftarrow$ **Gen**(1^k), every id_s and id_v, every $([a_s], x_s) \leftarrow$ **Ext**(x_m, pk_m, id_s), $([a_v], x_v) \leftarrow$ **Ext**(x_m, pk_m, id_v), then, the extractor can extract either x_s or x_v in expected time $t(\epsilon - \kappa)^{-1}$ with the help of the forger $\mathcal{F}$, where $\epsilon > \kappa$ without considering the required time to make oracle queries.

Definition 4. (Privacy of Signer's Identity) An IBSDVS scheme is $(t, q_{Ext}, q_{Sign}, q_{Sim}, q_{ver}, \epsilon)$-PSI-secure if no adversary $\mathcal{D}$ which runs in time at most t, issues at most q_{Sign} queries to $\mathcal{O}_{sign}$, q_{Sim} queries to $\mathcal{O}_{sim}$ and q_{ver} queries to $\mathcal{O}_{ver}$, can win the game bellow with probability that deviates from $1/2$ by more than ϵ.

The game is played between the challenger $\mathcal{C}$ and a distinguisher $\mathcal{D}$ as follows:

i. $\mathcal{C}$ runs **Gen** to generate a master key pair (x_m, pk_m), and gives pk_m to $\mathcal{D}$. Then $\mathcal{C}$ calls $\mathcal{O}_{Ext}$ for generating the key pairs $([a_{s_0}], x_{s_0})$, $([a_{s_1}], x_{s_1})$ and $([a_v], x_v)$ for S_0, S_1 and V with identities id_{s_0}, id_{s_1} and id_v, and invokes $\mathcal{D}$ on input $([a_{s_0}], [a_{s_1}], [a_v])$.
ii. $\mathcal{D}$ issues queries adaptively as in the unforgeability game, except that now all the oracles take an additional input $d \in \{0,1\}$ indicating which signer responds to the query. That is, the oracles generate

[1] By definition $\mathcal{D}ec_x\{\mathcal{E}nc_{[x]}\{w, r\}, [r]\} = w$

and verify signatures with respect to $[a_{s_d}]$ and $[a_v]$.

iii. $\mathcal{D}$ submits a message M^*. $\mathcal{C}$ tosses a coin $b \in \{0,1\}$, computes the challenge signature $\sigma^* \leftarrow \mathbf{Sign}(x_{s_b}, id_{s_b}, id_v, pk_m, M^*)$ and returns σ^* to $\mathcal{D}$.

iv. $\mathcal{D}$ continues to issue queries as in Step (ii). Finally it outputs a bit b' and wins the game if:
- $b' = b$; and
- it did not query $\mathcal{O}_{ver}$ on input (d, M^*, σ^*) for any $d \in \{0,1\}$.

2.4 Assumptions

The security proofs in this paper use the following assumptions:

Assumption 1 (one-wayness)- The one-wayness assumption holds for a group homomorphism if for all PPT Algorithms $\mathcal{A}$, the following probability:

$$\Pr(\forall x \in \mathcal{H} : x' \leftarrow \mathcal{A}([x]) : [x'] = [x]) \quad (10)$$

is a negligible function of the security parameter. In other words, it is hard to compute the preimage of a one-way homomorphism. In the case of the Schnorr-like homomorphism, this assumption is analogous to the DL assumption.

Assumption 2 (Diffie-Hellman)- The DH assumption holds for a function $\mathcal{DH}\{[x], y\}$ defined on a group homomorphism, if for all PPT algorithms $\mathcal{A}$ the following probability:

$$\Pr\big(\forall x, y \in \mathcal{H} : \mathcal{DH}\{[x], y\} \leftarrow \mathcal{A}([x], [y])\big) \quad (11)$$

is a negligible function of the security parameter.

Assumption 3 (Gap Diffie-Hellman)- The GDH assumption holds for a function $\mathcal{DH}\{[x], y\}$ defined on a group homomorphism, if for all PPT algorithms $\mathcal{A}$ having oracle access to $\mathcal{DDH}$ which on the inputs $[x]$, $[y]$ and Δ, correctly decides whether or not $\Delta = \mathcal{DH}\{[x], y\}$, the following probability:

$$\Pr\Big(\forall x, y \in \mathcal{H} : \mathcal{DH}\{[x], y\} \leftarrow \mathcal{A}^{\mathcal{DDH}(\cdot)}([x], [y])\Big) \quad (12)$$

is a negligible function of the security parameter.

3 Non-Delegatable SDVS without Pairings

In this section we present a model for a non-delegatable SDVS using trusted third party. The scheme is composed of a lightweight IBS and an efficient non-delegatable SDVS. These two building blocks are described in the next subsections. In the sequel of this section, the construction of our proposed scheme, together with its security proof in the random oracle model, will be presented.

3.1 Lightweight IBS

The identity based signature (IBS) scheme presented in (Galindo, and Garcia 2009) enjoys the property of not using pairings. Founding the signature only on the modular exponentiations and avoiding the high-cost time-taking pairing computations, Galindo and Garcia were able to propose a lightweight ID-based signature scheme in which two schnorr-like signatures are concatenated: one for the PKG (private key generator) to sign the identity of the signer and one for the signer to sign the message (Galindo, and Garcia 2009).

In fact, their novel idea is not limited to the schnorr-like signatures. With the help of Maurer's unified zero knowledge proof of knowledge together with the Fiat-Shamir heuristic (Fiat and Shamir 1987), the aforementioned Schnorr-like signature can be generalized to an abstract model for lightweight IBS.

Another efficient IBS scheme that is based on the elliptic curve discrete log problem and does not use bilinear pairings is suggested by Zhu, Yang, and Wong (2007). Nevertheless, these two IBS schemes are quite similar in the abstraction level we reviewed in section 2.1. The only difference between the two foregoing IBS schemes lies in the structure of the user's secret key where the scheme of Galindo, and Garcia (2009) releases the image of the random value selected in the key generation phase, while the hash output is released in the Zhu, Yang, and Wong (2007) scheme.

Figure 1 depicts this abstract model upon which we will set up our lightweight ID-based SDVS with non-delegatability. Although $[a_s]$ is included in the secret key of S, it is not really secret. Being included in the signature, $[a_s]$ is a commitment to ensure a verifier that the signature comes from id_s whose identity is signed by the PKG. A concrete construction of this scheme can be obtained by using a Schnorr-like homomorphism which is $[a_s] = g^{a_s}$, $pk_m = g^{x_m}$ and $R_s = g^{r_s}$.

Key Generation:	***Signing***:	***Verification***:
$(x_m, pk_m = [x_m]) \leftarrow \mathbf{Gen}(1^k)$	$\sigma \leftarrow \mathbf{Sign}(x_s, id_s, pk_m, M)$	$(b) \leftarrow \mathbf{Ver}(\sigma, id_s, pk_m, M)$
Key Extraction:	$\sigma = ([a_s], R_s, z_s, \sigma^+)$	$d_s = H_m(id_s, [a_s])$
$([a_s], x_s) \leftarrow \mathbf{Ext}(x_m, pk_m, id_s)$	$r_s \overset{\$}{\leftarrow} \mathcal{H}, R_s = [r_s]$	$X_s = [a_s] \otimes pk_m^{d_s} = [x_s]$
$\begin{cases} a_s \overset{\$}{\leftarrow} \mathcal{H}, d_s = H_m(id_s, [a_s]) \\ x_s = a_s \oplus (d_s \cdot x_m) \end{cases}$	$c_s = H_s(id_s, M, R_s, f(\sigma^+))$ $z_s = r_s \oplus (c_s \cdot x_s)$	$c_s = H_s(id_s, M, R_s, f(\sigma^+))$ $b = ([z_s] = R_s \otimes X_s^{c_s})\ ?\ 1 : 0$

Figure 1 Abstract model for lightweight IBS

Note that the signature σ in figure 1 contains some extra terms which are represented by σ^+ and have a contribution in generating the challenge c_s. This extra information can help adding some extra properties such as non-transferability, non-delegatability and privacy of the signer's identity to the signature scheme. This is further discussed in the following section.

3.2 Non-Delegatable SDVS

In order to construct a signature scheme which is strong designated verifier, non-transferable, non-delegatable and has PSI, Huang, Yang, Wong, and Susilo (2011a) suggested the following ideas:

I1- *Perfect non-transferability*: S and V should be able to produce the same signature on any message. This can be achieved through producing a composition of: 1) a zero knowledge proof of one's secret key; and 2) a simulation of a zero knowledge proof of the other's secret key. These two ZKPs should constitute a hierarchical order such that producing a valid simulation of a ZKP of both of the secret keys at the same time has a negligible probability.

I2- *Privacy of signature's identity*: the common key shared between S and V can be included into the message to be signed, so that no one can distinguish the signature without knowing this shared key.

I3- *Non-delegatability*: for S not to be able to delegate his signing capability, the signature contains a (non-interactive) ZKP of signer's secret key. Moreover, to prevent V from delegating his capability, each signature contains a fresh (non-interactive) challenge which can only be (non-interactively) responded by the knowledge of the verifier's secret key.

The above ideas were successfully combined in (Huang, Yang, Wong, and Susilo 2011a) and a SDVS scheme with non-delegatability was proposed which does not use pairings and is secure in the random oracle model. The signing and verifying is carried out via four and five modular exponentiations[1], respectively, and the signature consists of five elements in $\mathbb{Z}_q$ (Huang, Yang, Wong, and Susilo 2011a). Here we propose a marginal improvement to this signature. Relegated the role of one element from the signature to another and with the same number of modular exponentiations for signing and verifying, our modified signature consists of four elements (instead of five). This is further discussed in the next section.

3.3 Construction of the Scheme

Through the rest of this paper, it is assumed that the output of hash functions are from $\mathbb{Z}_q$ for some prime number q. Wherever the output of a hash function is treated as the input/output of the homomorphism (elements from $\mathcal{H}$ or $\mathcal{G}$), it is implicitly assumed that a suitable mapping $e: \mathbb{Z}_q \to \mathcal{H}$ or $e': \mathbb{Z}_q \to \mathcal{G}$ is used.

Here is the description of our signature scheme:

- *Key Generation*: According to the security parameter k, the key generation function generates a master secret key x_m and its corresponding master public key $pk_m = [x_m]$.
- *Key Extraction*: Taking as input the master secret key x_m, the master public key pk_m and the identity of the signer id_s (verifier id_v), the key extraction function outputs a commitment $[a_s]$ (respectively $[a_v]$) and a challenge response $x_s = a_s \oplus (d_s \cdot x_m)$ (respectively $x_v = a_v \oplus (d_v \cdot x_m)$).
- *Signing*: To sign a message M for a designated verifier id_v, the signer id_s uses her secret key x_s and the master public key pk_m and performs as follows: first, to simulate a ZKPK of the verifier's secret key (cf. **I1** in section 3.2), S chooses a random challenge c_v, a random challenge response $z_v = r_v \oplus (x_v \cdot c_v)$ and computes the corresponding simulated commitment $[r_v] = [z_v] \otimes [x_v]^{c_v}$. Next, in order to present a ZKPK of her owns secret key, she commits to a random r_s, computes her challenge $c_s = H_s(M, R_s, R_v, \dots) - c_v$, and produces the challenge response $z_s = r_s \oplus (x_s \cdot c_s)$. Finally, S outputs the signature which contains: PKG's commitment on her identity $[a_s]$, her commitment $[r_s]$ and challenge response z_s, the simulated challenge response z_v and an encryption of c_v with the verifier's public key.
- *Verification*: To verify a signature σ on a message M from a signer id_s, the verifier id_v uses his secret key x_v and the master public key pk_m and performs as follows: first, V decrypts E_v using his secret key together with R_s to obtain c_v. Next, from c_v, x_v and z_v he computes R_v to obtain the required inputs for the hash function, evaluate it and consequently obtain c_s. Finally, he simultaneously checks the validity of the signer's challenge response with respect to her commitment and the validity of the PKG's signature on the identity of the signer.

The scheme is depicted in figure 2. As was discussed in section 3.1, $[a_s]$ and $[a_v]$ are publicly known. If this is not the case a priori, S can ask either PKG or V for $[a_v]$. Since $[a_v]$ contains no secret information, its release causes no harm. A concrete construction can be obtained for this model by using a Schnorr-like homomorphism; i.e. letting $pk_m = g^{x_m}$, $[a_s] = g^{a_s}$, $R_s = g^{r_s}$, $R_v = g^{z_v} \cdot (g^{x_v})^{-c_v}$, $\mathcal{DH}\{[x_s], x_v\} = g^{x_s \cdot x_v}$ and so on.

As was mentioned in the previous section, our modification to the $SDVS_2$ in (Huang, Yang, Wong, and Susilo 2011a) makes its output length shorter, without increasing the computational complexity. To be concise, our modification to the Huang et al.'s $SDVS_2$ is twofold:

1) To prevent the verifier from delegating his capability in (Huang, Yang, Wong, and Susilo 2011a), for some $w \overset{\$}{\leftarrow} \mathbb{Z}_q$, $pk_v^w = g^{w x_v}$ is included

[1] The computation of $g^{x_s \cdot x_v}$ is omitted since it can be computed once and stored for the rest of transactions.

Key Generation:	***Signing***:	***Verification***:
$(x_m, pk_m = [x_m]) \leftarrow \textbf{Gen}(1^k)$	$\sigma \leftarrow \textbf{Sign}(x_s, id_s, id_v, pk_m, M)$	$(b) \leftarrow \textbf{Ver}(\sigma, x_v, id_s, id_v, pk_m, M)$
Key Extraction:	$\sigma = ([a_s], R_s, E_v, z_s, z_v)$	$d_s = H_m(id_s, [a_s])$
$([a_s], x_s) \leftarrow \textbf{Ext}(x_m, pk_m, id_s)$	$d_v = H_m(id_v, [a_v])$	$X_s = [a_s] \otimes pk_m^{d_s} = [x_s]$
$\begin{cases} a_s \overset{\$}{\leftarrow} \mathcal{H}, d_s = H_m(id_s, [a_s]) \\ x_s = a_s \oplus (d_s \cdot x_m) \end{cases}$	$X_v = [a_v] \otimes pk_m^{d_v} = [x_v]$	$c_v = \mathcal{D}ec_{x_v}\{E_v, R_s\}$
	$r_s, z_v \overset{\$}{\leftarrow} \mathcal{H}, c_v \overset{\$}{\leftarrow} \mathbb{Z}_q$	$R_v = [z_v] \otimes [x_v]^{-c_v}$
$([a_v], x_v) \leftarrow \textbf{Ext}(x_m, pk_m, id_v)$	$R_s = [r_s], R_v = [z_v] \otimes X_v^{-c_v}$	$K = \mathcal{DH}\{X_s, x_v\}$
$\begin{cases} a_v \overset{\$}{\leftarrow} \mathcal{H}, d_v = H_m(id_v, [a_v]) \\ x_v = a_v \oplus (d_v \cdot x_m) \end{cases}$	$E_v = \mathcal{E}nc_{X_v}\{c_v, r_s\}$	$c_s = H_s(id_s, id_v, M, R_s, R_v, K) - c_v$
	$\mathrm{K} = \mathcal{DH}\{X_v, x_s\}$	$b = ([z_s] = R_s \otimes X_s^{c_s})\ ?\ 1 : 0$
	$c_s = H_s(id_s, id_v, M, R_s, R_v, K) - c_v$	
	$z_s = r_s \oplus (c_s \cdot x_s)$	

Figure 2 Non-Delegatable SDVS using trusted third party

in the input of the hash function, while the signature contains g^w. Therefore, only with the knowledge of x_v one can obtain the output of the hash function and verify the signature. In the modified scheme, however, this role is played by E_v which, together with the R_s, constitute an ElGamal encryption of c_v using the public key of V. As a consequence, g^w is removed from the signature and the signature size is reduced from five elements to four elements. This idea in which an element is saved by sharing the random element $R_s = [r_s]$ between the ElGamal encryption E_v and the Schnorr signature component was used previously in Zheng's signcryption scheme (Zheng 1997).

2) The signature in (Huang, Yang, Wong, and Susilo 2011a) contains both the c_s and c_v and the verification is carried out by checking the equality of $c_s + c_v$ to the output of the hash function. Verifying the signature in the modified scheme is somehow different. With the R_s included in the signature, the verification is carried out much like the ZKPs. This introduces two extra exponentiations to the verification which, together with an exponentiation for decrypting E_v, results in three more exponentiations compared to the verification of the original scheme. However, the original scheme requires two exponentiations for evaluating R_s and one for g^{wx_v}. Since these three computations are omitted in the modified scheme, the efficiency of the verification remains the same to the original one.

Remark. The term g^{wx_v} plays no role in the security proofs of Huang et al.'s $SDVS_2$. Similarly, our scheme which replaces g^{wx_v} with $E_v = \mathcal{E}nc_{X_v}\{c_v, r_s\}$, does not depend on its security. The question is: "why this element which is supposed to prevent V from delegating her ability, does not appear in the security proof of non-delegatability (Theorem 3 in this paper and Theorem 6 in (Huang, Yang, Wong, and Susilo 2011a))?" This is because:

- **Simulating** a valid signature by V, requires not only the shared key ($K = \mathcal{DH}\{[x_s], x_v\}$), but also the private key of V (in order to compute the $z_v = r_v \oplus (x_v \cdot c_v)$). Therefore, if someone other than S generates a valid signature, showing that she knew x_v does not require the decryption of E_v; i.e. anyone who can compute z_v must have known x_v (and the knowledge of K alone does not suffice). In this case, as appears in the security proofs, inclusion of E_v instead of c_v in the signature has no impact in the security.
- **Verifying** a signature, on the other hand, can perfectly be accomplished using the knowledge of K alone, if c_v is known. Therefore, an encryption of c_v (which is E_v) is included in the signature. From this point of view, the role of g^{wx_v} in the $SDVS_2$ of (Huang, Yang, Wong, and Susilo 2011a) and $E_v = \mathcal{E}nc_{X_v}\{c_v, r_s\}$ in our scheme is exactly the same; i.e. they both prevent a third party to verify the signature without knowing x_v.

3.4 Security Analysis

The security of our scheme follows immediately from the security of its two components: the lightweight IBS (section 3.1) and the modified non-delegatable SDVS (section 3.2). However, for the sake of concreteness, we provide the security proofs of our model for the signature scheme. Our scheme is proved to have the desired properties in the random oracle model, provided that the one-wayness and the GDH assumptions hold.

Our scheme uses two hash functions H_m and H_s which are modeled as random oracles during the security proofs. As usual, these random oracles are simulated by keeping two separate lists L_m and L_s containing the queried values together with their corresponding answers. There are four other oracles involved in the security proofs of our scheme, as follows:

- **Key Extraction Oracle** ($\mathcal{O}_{Ext}$) which takes in an identity id and gives out $\mathsf{Ext}(x_m, pk_m, id)$.
- **Signing Oracle** ($\mathcal{O}_{sig}$) which takes in the three-tuple (id_s, id_v, M) and generates $\sigma = \mathsf{Sign}(x_s, id_s, id_v, pk_m, M)$
- **Simulation Oracle** ($\mathcal{O}_{sim}$) which simulates the $\mathcal{O}_{sig}$, using x_v instead of x_s.
- **Verification Oracle** ($\mathcal{O}_{ver}$) which on the input of (σ, id_s, id_v, M), determines whether or not σ is a valid SDVS of M from id_s to id_v.

Since the scheme is perfectly non-transferable (Theorem 2), the queries to $\mathcal{O}_{sim}$ need not be dealt with. The following four theorems formally prove the security of the signature scheme of figure 2 in the random oracle model. In what follows the scheme described in figure 2 is called IBSDVS and the homomorphic function which is used in the scheme is represented by f.

Theorem 1 (Unforgeability) – If one-wayness assumption (t_{ow}, ϵ_{ow})–holds for f, IBSDVS is $(t_{uf}, q_s, q_m, \epsilon_{uf})$–strongly unforgeable in the random oracle model, where $t_{ow} \approx t_{uf}$ and either $\epsilon_{ow} \geq \frac{\epsilon_{uf}}{2q_s}(\epsilon_{uf}/q_s^2 - 1/2^\eta)$ (unforgeability of the signing part) or $\epsilon_{ow} \geq \epsilon_{uf}(\epsilon_{uf}^3/(q_m q_s)^6 - 3/2^\eta)$ (unforgeability of the key extraction part), where η is the security parameter.

Proof As for all the IBS schemes, the unforgeability of our signature scheme is twofold: (I) the unforgeability of the Signing part; and (II) the unforgeability of the Key Extraction part. Assume $\mathcal{F}$ forges a valid signature (M^*, σ^*) on behalf of id^* with probability at most ϵ_{uf}, in time at most t_{uf}, making q_s, q_m queries to H_s and H_m. Based on the id^* two cases are distinguishable:

1) $\mathcal{F}$ forged a valid signature on behalf of id^* after querying $\mathcal{O}_{sig}$ for at least one signature with id^* either as the signer or the verifier. In this case, the signing part is forged.
2) Querying no signatures from/to id^* from $\mathcal{O}_{sig}$, $\mathcal{F}$ was successful at forging a valid signature on her behalf. This is the case where the key extraction part is forged.

To each case we will associate an adversary who, using $\mathcal{F}$ as a subroutine, can contradict the one-wayness assumption.

Case 1: In this case we will build $\mathcal{A}_1$ who finds a preimage of f with probability at least ϵ'_{ow}. With $\mathcal{O}_{sig}$ being forged by $\mathcal{F}$, $\mathcal{A}_1$ tries to extract the secret key of id^*. Note that in this case the signature from/to id^* on some message M is queried by $\mathcal{F}$, which means id^* is queried from H_m at least once. Let i^* represent the index of $\mathcal{F}$'s call to the random oracle H_m with the target identity id^*. Given a one-way homomorphism $f: \mathcal{G} \to \mathcal{H}$ (represented by $[\cdot]$) and an image $X = [x]$ for some unknown preimage $x \in \mathcal{G}$, $\mathcal{A}_1$ flips a fair coin to decide whether id^* is the identity of the forged signer, or the verifier. Here we assume the case where id^* corresponds to the forged signer. The simulation in the other case can be done similarly. $\mathcal{A}_1$ picks $i \overset{\$}{\leftarrow} \mathbb{Z}_{q_m}$, having a chance of $1/q_m$ at correctly guessing the target identity (i.e. $i = i^*$), and simulates the environment of $\mathcal{F}$ as follows:

Master Key Generation $\mathcal{A}_1$ chooses $x_m \overset{\$}{\leftarrow} \mathcal{H}$, sets $pk_m = [x_m]$, and uses (x_m, pk_m) as the master key pair for the $\mathcal{O}_{Ext}$ queries.

Hash Queries Given a query to H_m or H_s, if there is a tuple in L_m or L_s with the same input, $\mathcal{A}_1$ returns the corresponding answer; otherwise, it selects a fresh $c \overset{\$}{\leftarrow} \mathbb{Z}_q$, stores the new tuple in the appropriate list and returns c.

$\mathcal{O}_{Ext}$ Queries Every time $\mathcal{F}$ queries $\mathcal{O}_{Ext}$ for user id, $\mathcal{A}_1$ chooses $d \overset{\$}{\leftarrow} \mathbb{Z}_q, x' \overset{\$}{\leftarrow} \mathcal{H}$, sets $A = [x'] \otimes pk_m^{-d}$ and adds $\big((id, A), d\big)$ to the L_m. Then it returns (A, x') to $\mathcal{F}$.

$\mathcal{O}_{sig}$ Queries On input a message M, when the signer's identity differs from id^*, $\mathcal{A}_1$ simply computes the private key of S and Vas described in the $\mathcal{O}_{Ext}$ Queries and returns the output of the signing algorithm to $\mathcal{F}$. When id^* is the signer's identity, $\mathcal{A}_1$ chooses $d^* \overset{\$}{\leftarrow} \mathbb{Z}_q$, sets $A_s = X \otimes pk_m^{-d^*}$, adds $\big((id^*, A_s), d^*\big)$ to the L_m, and computes the private key of Vas described in the $\mathcal{O}_{Ext}$ Queries. It chooses $r_v, z_s \overset{\$}{\leftarrow} \mathcal{H}$ and $c_s, c \overset{\$}{\leftarrow} \mathbb{Z}_q$, computes $R_v = [r_v]$, $R_s = [z_s] \otimes X^{-c_s}$, $c_v = c - c_s$, $z_v = r_v \oplus (c_v \cdot x_v)$ and adds $\big((id^*, id_v, M, R_s, R_v, \mathcal{DH}\{X, x_v\}), c\big)$ to the L_s. Then from x_v and $R_s = [r_s]$, it computes $E_v = \mathcal{E}nc_{x_v}\{c_v, r_s\}$ and returns $\sigma = (A_s, R_s, E_v, z_s, z_v)$ to $\mathcal{F}$.

Finally, $\mathcal{F}$ outputs its forgery (M^*, σ^*) on behalf of id^* to id_v. If $\sigma^* = (A_s^*, R_s^*, E_v^*, z_s^*, z_v^*)$ is not valid, $\mathcal{A}_1$ aborts. Otherwise, $\mathcal{A}_1$ computes $c_v^* = \mathcal{D}ec_{x_v}\{E_v^*, R_s^*\}$ and $R_v^* = [z_v] \otimes [x_v]^{-c_v^*}$. Let c^* be the corresponding answer to the query $(id^*, id_v, M^*, R_s^*, R_v^*, \mathcal{DH}\{X, x_v\})$ which is stored in the L_s. $\mathcal{A}_1$ computes $c_s^* = c^* - c_v^*$ and rewinds $\mathcal{F}$ to the status where it queried H_s for $(id^*, id_v, M^*, R_s^*, R_v^*, \mathcal{DH}\{X, x_v\})$ and this time feeds $\mathcal{F}$ with $\bar{c}^* \neq c^*$. The subsequent queries are answered as above. At this stage, we assume $\mathcal{F}$ outputs $(M^*, \bar{\sigma}^*)$ which is another valid forgery on the same message. From the new signature $\bar{\sigma}^* = (A_s^*, R_s^*, \bar{E}_v^*, \bar{z}_s^*, \bar{z}_v^*)$, $\mathcal{A}_1$ computes $\bar{c}_v^* = \mathcal{D}ec_{x_v}\{\bar{E}_v^*, R_s^*\}$ and $\bar{c}_s^* = \bar{c}^* - \bar{c}_v^*$. If $\bar{c}_s^* = c_s^*$, $\mathcal{A}_1$ aborts. Otherwise, it outputs the preimage of X as follows:

$$[z_s^*] \otimes X^{-c_s^*} = R_s^* = [\bar{z}_s^*] \otimes X^{-\bar{c}_s^*} \Rightarrow [z_s^* \ominus \bar{z}_s^*] = [x]^{c_s^* - \bar{c}_s^*} = [(c_s^* - \bar{c}_s^*) \cdot x]$$

$$x = (z_s^* \ominus \bar{z}_s^*)/(c_s^* - \bar{c}_s^*) \quad (13)$$

Probability Analysis $\mathcal{A}_1$ has to make two guesses before using the forgeries of $\mathcal{F}$ for computing the preimage of f. First it guesses the id^* was queried at the ith call to the H_s with a probability of $1/q_s$. Secondly, it guesses with a $1/2$ probability that id^* belongs to the signer or the verifier of the forged signature. Conditioned on these two guesses, by an analysis similar to that in (Galindo, and Garcia 2009), the success probability of $\mathcal{A}_1$ can be

obtained by the Forking Lemma (Pointcheval, and Stern 2000) to be greater than:

$$\epsilon_{uf}\left(\frac{\epsilon_{uf}}{q_s^2}-\frac{1}{2^\eta}\right) \quad (14)$$

where η is the security parameter. Therefore, we have that:

$$\epsilon'_{ow} \geq \frac{\epsilon_{uf}}{2q_s}\left(\frac{\epsilon_{uf}}{q_s^2}-\frac{1}{2^\eta}\right) \quad (15)$$

Case 2: In this case we will build $\mathcal{A}_2$ who finds a preimage of f with probability at least ϵ''_{ow}. As was mentioned earlier, in this case the $\mathcal{O}_{Ext}$ is forged by $\mathcal{F}$. Consequently, $\mathcal{A}_2$ exploits this forgery to extract x from $[x]$. Given a one-way homomorphism $f: \mathcal{G} \to \mathcal{H}$ (represented by $[\cdot]$) and an image $X = [x]$ for some unknown preimage $x \in \mathcal{G}$, $\mathcal{A}_2$ sets $pk_m = X$ and tries to obtain $x_m = x$ by simulating the environment of $\mathcal{F}$ as follows:

$\mathcal{A}_2$ acts just the same as $\mathcal{A}_1$ did in *Case 1* until it obtains two valid forgeries (M^*, σ^*) and $(M^*, \bar{\sigma}^*)$ on behalf of id_s to id_v whose identities were never queried from H_m by $\mathcal{F}$. However, the following relation should be satisfied:

$$[x_s] = A_s^* \otimes X^{-d_s};\ \text{where}\ d_s \triangleq H_m(id_s, A_s^*) \quad (16)$$

$\mathcal{A}_2$ repeats the above process to obtain two more valid forgeries (M^*, σ') and $(M^*, \bar{\sigma}')$ on behalf of id_s to id_v; but this time it defines $d'_s \triangleq H_m(id_s, A_s^*)$ with $d'_s \neq d_s$. Ultimately, $\mathcal{A}_2$ is left with the following two equations:

$$\begin{cases} 1: \quad [z_s^*] \otimes X_s^{-c_s^*} = R_s^* = [\bar{z}_s^*] \otimes X_s^{-\bar{c}_s^*} \\ \quad \Rightarrow \ x_s = (z_s^* \ominus \bar{z}_s^*)/(c_s^* - \bar{c}_s^*) \\ 2: [z'_s] \otimes (X'_s)^{-c'_s} = R'_s = [\bar{z}'_s] \otimes (X'_s)^{-\bar{c}'_s} \\ \quad \Rightarrow \ x'_s = (z'_s \ominus \bar{z}'_s)/(c'_s - \bar{c}'_s) \end{cases} \quad (17)$$

and from the x_s and x'_s, the unknown preimage x is obtained as follows (note that A_s^* remains unchanged in all of the four forged signatures):

$$x_s \ominus x'_s = (d_s \cdot x) \ominus (d'_s \cdot x) \ \Rightarrow \ x = (x_s \ominus x'_s)/(d_s - d'_s) \quad (18)$$

Probability Analysis In this case, no guesses are made by $\mathcal{A}_2$ and by an analysis similar to that of (Galindo, and Garcia 2009), using the Multiple-Forking Lemma [BPW03, BN06] the success probability of $\mathcal{A}_2$ at finding a preimage of f is obtained as follows:

$$\epsilon''_{ow} \geq \epsilon_{uf}\left(\frac{\epsilon_{uf}^3}{(q_m q_s)^6}-\frac{3}{2^\eta}\right) \quad (19)$$

This completes the proof. ■

Theorem 2 (Non-Transferability) – IBSDVS is perfectly non-transferable.

Proof The subsequent steps can easily be followed by V in order to simulate the signature of S on M:

1. Compute: $d_s = H_m(id_s, [a_s]), X_s = [a_s] \otimes pk_m^{-d_s} = [x_s]$
2. Choose: $r_v, z_s \overset{\$}{\leftarrow} \mathcal{H}$ and $c_s \overset{\$}{\leftarrow} \mathbb{Z}_q$
3. Compute: $R_v = [r_v], R_s = [z_s] \otimes X_s^{-c_s}$
4. Compute: $c_v = H_s(id_s, id_v, M, R_s, R_v, \mathcal{DH}\{x_v, X_s\}) - c_s$
5. Using the knowledge of $(x_v, R_s = [r_s])$, compute: $E_v = \mathcal{E}nc_{X_v}\{c_v, r_s\}$
6. Compute: $z_v = r_v \oplus (x_v \cdot c_v)$
7. Output: $\sigma = ([a_s], R_s, E_v, z_s, z_v)$

It can be easily verified that the distribution of the ZKPK by V in the above steps is identical to the distribution of the ZKPK by S. ■

Theorem 3 (Non-Delegatability) – IBSDVS is non-delegatable with knowledge error $1/q$ in the random oracle model.

Proof Let $\mathcal{F}$ be the forger to whom the machine $\mathcal{K}$ has oracle access in order to produce either x_s or x_v. Let $\mathcal{F}_M$ be $\mathcal{F}$ with M as input, who forges a valid signature (M, σ) with $\sigma = ([a_s], R_s, E_v, z_s, z_v)$ on behalf of id_s to id_v, with probability ϵ after at most q_s queries from the H_s oracle. $\mathcal{K}$ executes $\mathcal{F}_M$ to the point where it outputs its forgery on M. Then it rewinds $\mathcal{F}_M$ to the status of asking the H_s oracle for $(id_s, id_v, M, R_s, R_v, \mathcal{DH}\{x_v, X_s\})$, replaces its answer c with a different value $\bar{c}$, and continues the execution of $\mathcal{F}_M$ until it outputs another signature $\bar{\sigma} = ([a_s], R_s, \bar{E}_v, \bar{z}_s, \bar{z}_v)$ on M. Since (R_s, R_v) are in the hash input, they remain the same in both runs. Furthermore, due to the following discussions, one of the pairs of (c_s, z_s) or (c_v, z_v) must remain unchanged: As was mentioned during I1 in section 3.2, it is impossible to generate a valid signature by selecting both c_s and c_v after obtaining $c = H_s(\dots, R_s, R_v, \dots)$; the construction of the scheme is such that the chronological order for computing the three parameters (c, c_s, c_v) is forced to be either (c_s, c, c_v) or (c_v, c, c_s). Here, as one of c_s or c_v was selected prior querying H_s, changing the answer of H_s oracle only affects one of c_v or c_s. Therefore, either:

$$c_s = \bar{c}_s \Rightarrow c - \bar{c} = c_v - \bar{c}_v \quad (20)$$

or

$$c_v = \bar{c}_v \Rightarrow c - \bar{c} = c_s - \bar{c}_s \quad (21)$$

In the first case, $\mathcal{K}$ computes $x_v = (z_v \ominus \bar{z}_v)/(c - \bar{c})$; while the second case leads to $x_s = (z_s \ominus \bar{z}_s)/(c - \bar{c})$.

Note that the extractor $\mathcal{K}$ does not know the value of $\mathcal{DH}\{x_v, X_s\}$ and it guesses in which hash query the last element is the correct value of $\mathcal{DH}\{x_v, X_s\}$. This imposes a factor of $1/q_s$ to the extractor's success probability. By a similar analysis to that in (Boneh, Boyen, and Shacham 2004, and Pointcheval, and Stern 2000), $\mathcal{K}$ succeeds at extracting either x_s or x_v with a probability at least $\frac{\epsilon - 1/q}{16q_s}$.

Theorem 4 (PSI) – if GDH assumption $(t_{gdh}, \epsilon_{gdh})$-holds, IBSDVS is then (t, ϵ)-PSI-secure in the random oracle model, where $t \approx t_{gdh}$ and $\epsilon \leq \epsilon_{gdh} + 2/q$.

Proof Let $\mathcal{D}$ be an adversary who can break the privacy of signer's identity with probability $\frac{1}{2}+\epsilon$ in time at most t. We build an algorithm $\mathcal{E}$ which uses $\mathcal{D}$ as a subroutine to solve the GDH problem. The idea is that $\mathcal{D}$ cannot distinguish between the signatures of S_0 and S_1 unless it had queried (or otherwise correctly guessed the output of) the random oracle H_s for $\mathcal{DH}\{X_v, x_{s_0}\}$ or $\mathcal{DH}\{X_v, x_{s_1}\}$. $\mathcal{E}$ monitors $\mathcal{D}$'s calls to H_s for the aforementioned query and succeeds in solving the GDH problem as soon as $\mathcal{D}$ makes such a query.

Assume $\mathcal{E}$ is given $U_0=[u_0]$, $U_1=[u_1]$, $W=[w]$ and a DDH oracle and it wants to compute either $\mathcal{DH}\{U_0, w\}$ or $\mathcal{DH}\{U_1, w\}$. $\mathcal{E}$ is successful if it outputs (K, b) where $K=\mathcal{DH}\{U_b, w\}$. To do so, it sets $X_{s_0}=U_0$, $X_{s_1}=U_1$ and $X_v=W$ for S_0, S_1 and V with identities id_{s_0}, id_{s_1} and id_v. It also selects $x_m \overset{\$}{\leftarrow} \mathcal{H}$ and sets $pk_m=[x_m]$ for the master public and private keys. It then selects $d_{s_0}, d_{s_1}, d_v \overset{\$}{\leftarrow} \mathbb{Z}_q$, computes $A_{s_0}=U_0 \otimes pk_m^{-d_{s_0}}$, $A_{s_1}=U_1 \otimes pk_m^{-d_{s_1}}$ and $A_v=W \otimes pk_m^{-d_v}$ and adds $\left(\left(id_{s_0}, A_{s_0}\right), d_{s_0}\right)$, $\left(\left(id_{s_1}, A_{s_1}\right), d_{s_1}\right)$ and $\left((id_v, A_v), d_v\right)$ to the L_m. $\mathcal{E}$ simulates the environment of $\mathcal{D}$ as follows:

Hash Queries On the input $\left(id_{s_b}, id_v, M, R_s, R_v, K\right)$, if this was queried before, $\mathcal{E}$ returns the corresponding answer. Otherwise, using the $\mathcal{DDH}$ oracle, it checks whether $K=\mathcal{DH}\{X_v, x_{s_b}\}$. If this equation holds, $\mathcal{E}$ successfully terminates the run by returning (K, b); else it selects $c \overset{\$}{\leftarrow} \mathbb{Z}_q$ and appends the new tuple to the L_s.

$\mathcal{O}_{sig}$ Queries Given a bit b and a message M, $\mathcal{E}$ chooses $E_v, R_s \overset{\$}{\leftarrow} \mathcal{G}$, $z_s, z_v \overset{\$}{\leftarrow} \mathcal{H}$ and returns $\left(A_{s_b}, E_v, R_s, z_s, z_v\right)$. Note that decrypting a random E_v only with the knowledge of X_v and R_s implies breaking the DH assumption. Furthermore, in order to obtain c_s, one should know the decryption of E_v together with c. Consequently, checking the distribution of a signature without knowing the private key of the signer of verifier (to decrypt E_v), contradicts the DH assumption. This means $\mathcal{D}$ cannot distinguish between a real signature and the one which $\mathcal{E}$ provided.

$\mathcal{O}_{ver}$ Queries Given a bit b, a message M and an alleged signature σ, if the signature was generated before, $\mathcal{E}$ returns 1; otherwise, it returns 0. As there is a $1/q$ probability that $\mathcal{D}$ correctly guesses the answer of H_s without querying it and successfully forges a valid signature, this behavior of $\mathcal{E}$ reduces the success probability of $\mathcal{D}$ at winning the game by $1/q$.

When it is ready, $\mathcal{D}$ submits a challenge message M^*. $\mathcal{E}$ then selects $b \overset{\$}{\leftarrow} \{0,1\}$ and runs the $\mathcal{O}_{sig}$ as was discussed in the $\mathcal{O}_{sig}$ Queries. Then $\mathcal{E}$ continues to simulate oracles for $\mathcal{D}$ and monitor the hash queries as above. Finally, if the run has not been terminated until now, $\mathcal{D}$ outputs a bit b' and $\mathcal{E}$ aborts.

Probability Analysis By a discussion like that of (Huang, Yang, Wong, and Susilo 2011a), there is at least a $1-1/q$ probability that $\mathcal{D}$ wins the game without querying H_s for either $\mathcal{DH}\{X_v, x_{s_0}\}$ or $\mathcal{DH}\{X_v, x_{s_1}\}$, $1/q$ being the probability of correctly guessing the output of H_s. Furthermore, as was shown during the security analysis, there is a $1/q$ difference in the simulation of the $\mathcal{O}_{ver}$. Therefore, ϵ can be bounded by:

$$\epsilon_{gdh} \geq \left(1-\frac{1}{q}\right)\left(\epsilon-\frac{1}{q}\right) > \epsilon-\frac{2}{q} \qquad (22)$$

This completes the proof. ■

3.5 Comparison to Previous Schemes

A comparison between the efficiency of previous IB(S)DVS schemes and our proposal is shown in table 1. It should be noted that all of these schemes are provably secure in the random oracle model. In this table, the dominating computational signing and verification cost as well as the signature size are compared in the existing IB(S)DVS schemes. The table also contains two additional columns ND and PSI, respectively indicating whether the scheme supports non-delegatability and privacy of signer's identity.

4 Conclusions and Future Work

We proposed a non-delegatable IBSDVS scheme without pairings and proved its security in the random oracle model. To the best of our knowledge, this is the first non-delegatable IBSDVS which can be efficiently employed in the resource constraint applications since it has smaller signature size compared to previous non-delegatable IBSDVS and does not suffer the costly computations imposed by the use of pairings.

Table 1. Comparison between our scheme and the previous IB(S)DVS schemes

Scheme	Signing Cost	Verification Cost	Signature size	ND	PSI
Huang, Susilo, Mu, and Zhang (2008)	$1P$	$1P$	$1Z_p$	X	√
Kang, Boyd, and Dawson (2009)	$2P+2E+1E_T$	$1P+1E_T$	$2G_T$	X	√
Zhang, and Mao (2008)	$4E$	$3P$	$3G$	X	√
Huang, Susilo, and Wong (2009)	$3P+1E+3E_T$	$4P+4E_T$	$1G+4Z_p$	√	X
Huang, Yang, Wong, and Susilo (2011b)	$3P+2E+4E_T$	$5P+1E+4E_T$	$2G+2G_T+3Z_p$	√	√
Chow (2008)	$1P$	$2P+1E$	$2G$	X	X
Rajabzadeh Asaar, Salmasizadeh (2012)	$4E$	$6E$	$4Z_p$	√	X
Ours	$6E$	$7E$	$5G$	√	√

The proposed scheme can be instantiated for any one-way group homomorphism where GDH assumption is believed to be true. As a consequence, the literature can be investigated for one-way homomorphisms with a desired property in order to make the scheme more efficient or stronger than that which is implemented using a Schnorr-like homomorphism. A good example for such a case may be the use of lattice-based homomorphic functions in order to make the scheme a post-quantum signature.

5 References

Bellare, M., Namprempre, C., Neven, G. (2004): Security proofs for identity-based identification and signature schemes. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 268–286. Springer, Heidelberg.

Boneh, D., Boyen, X. and Shacham, H. (2004): Short group signatures, In: Proceedings of Advances in Cryptology—CRYPTO 2004, vol. 3152 of LNCS, pp. 41–55, Springer.

Cao, F. and Cao, Z. (2009): An identity based universal designated verifier signature scheme secure in the standard model, International Journal of Systems and Software, vol. 82(4), pp. 643-649.

Cha, J.C. and Cheon, J.H. (2002): An identity-based signature from gap Diffe-Hellman groups. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 18–30. Springer, Heidelberg.

Chow, S. (2008): Multi-Designated Verifiers Signatures Revisited. I. J. Network Security 7(3): pp. 348-357.

Chow S. (2006): Identity-Based Strong Multi-Designated Verifiers Signatures. EuroPKI 2006: pp. 257-259

Diffie, W. and Hellman, M.E. (1976): New directions in cryptography, IEEE Transactions on Information Theory 22(6), pp. 644–654.

ECRYPT, Ecrypt yearly report on algorithms and key length (2006): revision 1.1, http://www.ecrypt.eu.org/documents/D.SPA.21-1.1.pdf.

ElGamal, T. (1984): A public key cryptosystem and a signature scheme based on discrete logarithms, In G. R. Blakley and D. Chaum (Ed.), CRYPTO'84, LNCS 196, pp. 10–18, Springer.

Fiat, A. and Shamir, A. (1987): How to prove yourself: practical solutions of identification and signature problems, In: A.M. Odlyzko (Ed.) CRYPTO'86, LNCS 263, pp. 186–194, Springer-Verlag.

Galindo, D. and Garcia, F. D. (2009): A Schnorr-Like Lightweight Identity-Based Signature Scheme, In: B. Preneel (Ed.) AFRICACRYPT'09, LNCS 5580, pp. 135-148, Springer-Verlag.

Guillou, L.C. and Quisquater, J.J. (1988): A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory, In: C.G. Günther (Ed.) EUROCRYPT'88, LNCS 330, pp. 123–128, Springer, Heidelberg, 1988.

Guillou, L.C. and Quisquater, J.J. (1990): A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 216–231. Springer, Heidelberg.

Hess, F. (2003): Effcient identity based signature schemes based on pairings. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg.

Huang, X., Mu, Y., Susilo, W. and Wu, W. (2007): Provably secure pairing-based convertible undeniable signature with short signature length. In: Proceedings of 1st International Conference on Pairing-Based Cryptography, Pairing 2007, vol. 4575 of Lecture Notes in Computer Science, pp. 367–391. Springer.

Huang, X., Susilo, W., Mu, Y. and Zhang, F. (2008): Short designated verifier signature scheme and its identity- based variant, International Journal of Network Security, vol. 6(1), pp.82-93.

Huang, Q., Susilo, W., Wong, D. S. (2009): Non-delegatable identity-based designated verifier signature, Cryptology ePrint Archive, Report 2009/367.

Huang, Q., Yang, G., Wong, D. S. and Susilo, W. (2011a): Efficient strong designated verifier signature schemes without random oracle or with non-delegatability, International Journal of Security, pp. 373-385.

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

Jakobsson, M., Sako, K. and Impagliazzo, R. (1996): Designated verifier proofs and their applications. In: Proceedings of Advances in Cryptology—EUROCRYPT 1996, vol. 1070 of Lecture Notes in Computer Science, pp. 143 – 154. Springer.

Kang, B., Boyd, C. and Dawson, E. (2009): A novel identity based strong designated verifier signature scheme, International Journal of Systems and Software, vol. 82(2), pp. 270-273.

Laguillaumie, F. and Vergnaud, D. (2004): Designated verifier signatures: anonymity and efficient construction from any bilinear map. In: Proceedings of 4th International Conference on Security in Communication Networks, SCN 2004, vol. 3352 of Lecture Notes in Computer Science, pp. 105–119. Springer

Lipmaa, H., Wang, G. and Bao, F. (2005): Designated verifier signature schemes: Attacks, new security notions and a new construction. In: Proceedings of 32th International Colloquium on Automata, Languages and Programming, ICALP 2005, LNCS 3580 , pp. 459–471. Springer.

Maurer, U. (2009): Unifying Zero Knowledge Proofs of Knowledge, In: B. Preneel (Ed.) AFRICACRYPT'09, LNCS 5580, pp. 272-286, Springer-Verlag, 2009.

Okamoto, T. (1993): Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg.

Pointcheval, D. and Stern, J. (2000): Security arguments for digital signatures and blind signatures, Journal of Cryptology 13(3), pp. 361–396.

Rajabzadeh Asaar, M., Salmasizadeh, M. (2012): A Non-delegatable Identity-based Designated Verifier Signature Scheme without Bilinear Pairings. IACR Cryptology ePrint Archive (IACR) 2012:332

Sakai, R., Ohgishi, K. and Kasahara, M. (2000): Cryptosystems based on pairing. In: The 2000 Symposium on Cryptography and Information Security, Oiso, Japan.

Shamir, A. (1985): Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg.

Schnorr, C. P. (1991): Effcient signature generation by smart cards. Journal of Cryptology 4(3), pp. 161–174.

Wang, B. and Song, Z. (2009): A non-interactive deniable authentication scheme based on designated verifier proofs. Inf. Sci. 179(6), pp. 858–865.

Zhang, J. and Mao, J. (2008): A novel id-based designated verifier signature scheme, International Journal of Information Sciences, vol.178(3), pp.766-773.

Zheng, Y. (1997): Digital signcryption or how to achieve Cost (Signature & Encryption) << Cost (Signature) + Cost (Encryption), Advances in Cryptology–CRYPTO'97, vol. 1294 of Lecture Notes in Computer Science, pp.165-179, Springer-Verlag.

Zhu, R., Yang G. and Wong, D. S. (2007): An Efficient Identity-Based Key Exchange Protocol with KGS Forward Secrecy for Low-Power Devices, Theoretical Computer Science, 378(2), pp. 198-207.

Appendix A Some Constructions for GQ-Like Homomorphism

A function for $\mathcal{DH}\{[x_s], x_v\}$, in the case of a GQ-like homomorphism, can be achieved as follows:

- The entity in charge of the key distribution (KD) chooses a secret $\alpha \in \mathbb{Z}_n^*$ with order $\varphi(m)$. In order to set $r \overset{\$}{\leftarrow} \mathbb{Z}_m^*$ in his transactions, KD randomly selects $\zeta \overset{\$}{\leftarrow} \mathbb{Z}_n^*$ instead, and sets $r = \alpha^{\zeta} \bmod n$.
- At the end of the key generation procedure for the party P, KD sends to P the pair of $(\xi_p, x_p = \alpha^{\xi_p} \bmod n)$ as his secret key[1].
- In order to compute their common key, S and V can each do as follows:

$$\mathcal{DH}\{[x_s], x_v\} = [x_s]^{\xi_v} = \alpha^{e\xi_s\xi_v} = [x_v]^{\xi_s} = \mathcal{DH}\{[x_v], x_s\} \quad (23)$$

The secrecy of α is required if the one-wayness is going to be independent of the discrete logarithm assumption.

It should be noted that the above construction needs more analysis and evaluation.

[1] Note that obtaining α from x_p means computing the ξ_p-th roots which is believed to be infeasible.