On Delegatability of Four Designated Verifier Signatures

Yong Li[1], Helger Lipmaa[2,3], and Dingyi Pei[1]

[1] State Key Laboratory of Information Security (Graduate School of Chinese Academy of Sciences), Beijing 100049, P.R.China
[2] Cybernetica AS, Lai 6, 51005 Tartu, Estonia
[3] Institute of Computer Science, University of Tartu, J. Liivi 2, 50409 Tartu, Estonia

Abstract. In a paper recently published in ICALP 2005, Lipmaa, Wang and Bao identified a new essential security property, non-delegatability, of designated verifier signature (DVS) schemes. Briefly, in a non-delegatable DVS scheme, neither a signer nor a designated verifier can delegate the signing rights to any third party T without revealing their secret keys. We show that the Susilo-Zhang-Mu identity-based strong DVS scheme, Ng-Susilo-Mu universal designated multi verifier signature scheme, the Laguillaumie-Vergnaud multi-DVS scheme and the Zhang-Furukawa-Imai universal DVS scheme are delegatable. Together with the results of Lipmaa, Wang and Bao, our results show that most of the previously proposed DVS schemes are delegatable. However, the Laguillaumie-Vergnaud and Zhang-Furukawa-Imai schemes may still be secure in practice, since there the only party who can delegate signing is the designated verifier, who may not have motivation to do so. We finish the paper with some discussion on whether the non-delegatability notion of Lipmaa, Wang and Bao is appropriate.
Keywords. Designated verifier signatures, non-delegatability.

1 Introduction

A designated verifier signature (DVS) scheme [JSI96,Cha96] enables a signer to sign a message so that the designated verifier can verify it as coming from the signer. However, the designated verifier cannot transfer the conviction to others because he himself is able to generate signatures according to a distribution that is computationally or statistically close to the distribution of signatures, generated by the signer. On the other hand, nobody else but the signer and the designated verifier can generate valid signatures.

Recently, Lipmaa, Wang and Bao revisited the DVS security in [LWB05]. In particular, they identified a new security property for DVS, non-delegatability, and showed that several previously proposed DVS schemes [SKM03,SBWP03,SWP04,LV04a] are delegatable. Informally speaking, DVS is delegatable if either the signer or the designated verifier can delegate the signing rights (either with respect to a concrete designated verifier or with respect to all designated verifiers) to some third party T without disclosing his or her secret key. Delegatability, especially with respect to a concrete designated verifier, is highly undesirable in many applications. For example, in an e-voting scenario where a voter signs messages by using a delegatable DVS scheme (with the tallier being the designated verifier), one voter can delegate her voting right to a

coercer that can then vote instead of the voter. Therefore, such an e-voting protocol is coercible. Moreover, in many e-commerce applications, one can use a DVS scheme so that the signer is a subscriber to an e-service provided by a service provider who is the designated verifier. If the DVS scheme is delegatable, signer can send some delegation token to a non-subscriber who can then enjoy the service for free.

Our contributions. In addition to the negative results of [LWB05], we show that the Susilo-Zhang-Mu ID-based DVS scheme SZM04 [SZM04], the Ng-Susilo-Mu universal designated multi verifier signature scheme NSM05 [NSM05], the Zhang-Furikawa-Imai DVS scheme ZFI05 [ZFI05] and the Laguillaumie-Vergnaud MDVS scheme LV04 [LV04b] are delegatable. Together with [LWB05], our results show that almost all DVS schemes in the literature are delegatable. In particular, all DVS schemes based on bilinear maps are delegatable. The only non-delegatable DVS schemes that we are aware of are the schemes from [JSI96,LWB05] that are both built by using standard proof of knowledge techniques.

All delegation attacks, proposed in [LWB05], had a similar structure: they showed that either the signer or the designated verifier can delegate the signing rights by publishing some Diffie-Hellman key. Our attacks are more varied. In particular, our attacks against ZFI05 and LV04, while being attacks according to the definitions of Lipmaa, Wang and Bao, might sometimes be not very serious in practice. Namely, in both cases, only the designated verifier (in the case of LV04, the coalition of two designated verifiers) can delegate the signing. This means that attacks in the scenarios, outlined above, will not work. However, there are still some possibilities of cheating. For example, the service provider can forward the delegation token to a third party, who can then use the service indistinguishably from the real signer. By doing so, the third party could act in a way that ruins the reputation of the real signer. Whether this is a real attack or not, depends on the situation; we will leave it as an open question. For applications where this is a real attack, one should not use ZFI05 and LV04. In applications where it is not, one should give an alternative and possibly weaker definition of non-delegatability than was done in [LWB05].

Inspired on this, we give an informal definition of a weaker notion of delegatability that we call *verifier-only delegatability*. Intuitively, a (multi-)DVS scheme is verifier-only delegatable if the designated verifier (but not the signer) can delegate the signing rights to some third party. Clearly, a verifier-only delegatable DVS scheme is also delegatable. It seems (seems, since we are not going to give proofs that the signer cannot delegate) that the LV04 and the ZFI05 schemes are verifier-only delegatable.

Moreover, the presented attacks can also be divided into delegation attacks that allow to delegate the signing rights of a fixed signer w.r.t. a fixed tuple of designated verifiers, or the rights of *any* signer w.r.t. a fixed tuple of designated verifiers, or the rights of a fixed signer w.r.t. any tuple of designated verifiers. According to [LWB05], existence of any of these attacks makes a scheme delegatable. Again, it can be argued that the first type of attack is the most serious one since the last two attack types give too much power to the third party T (and therefore one might be less motivated to delegate the rights). One can try to modify the delegatability definition so that only the

first attack type is classified as an attack. We will leave it as an open question whether this is reasonable.

Regardless of the previous comments, our own opinion is that our attacks against all four schemes indicate some weaknesses in them and while the non-delegatability definition of [LWB05] might be too strong in some sense, to avoid any kind of future attack and unexpected vulnerabilities, it is a good idea to design DVS schemes that are non-delegatable according to [LWB05].

Road-map. Formal definition of an n-DVS scheme and its security is given in Sect. 2. In Sect. 3, we review four DVS schemes and present our delegation attacks against every single one of them. We discuss the different delegation attacks and define the novel notion verifier-only non-delegatability in Sect. 4.

2 Preliminaries

Let $\mathbb{G}$ be a cyclic additive group generated by P, whose order is a prime q, and let $\mathbb{H}$ be a cyclic multiplicative group of the same order q. A bilinear pairing is a map $\langle \cdot, \cdot \rangle : \mathbb{G} \times \mathbb{G} \to \mathbb{H}$ with the following properties:

Bilinearity: $\langle aP, bQ \rangle = \langle P, Q \rangle^{ab}$ for all $P, Q \in \mathbb{G}$ and $a, b \in \mathbb{Z}_q^*$;
Non-degeneracy: There exist $P, Q \in \mathbb{G}$ such that $\langle P, Q \rangle \neq 1$;
Computability: There is an efficient algorithm to compute $\langle P, Q \rangle$ for all $P, Q \in \mathbb{G}$.

Formal Definition of n-DVS. Next, we present a formal definition of n-DVS for $n \geq 1$, generalising the definition from [LWB05]. Let S be the signer, and $D_1, \ldots, D_n$ be n designated verifiers. In the following, we will denote $(\mathsf{PK}_{D_1}, \ldots, \mathsf{PK}_{D_n})$ by $\mathsf{PK}_{\boldsymbol{D}}$, $(\mathsf{SK}_{D_1}, \ldots, \mathsf{SK}_{D_n})$ by $\mathsf{SK}_{\boldsymbol{D}}$, and $(\mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{D_1}}, \ldots, \mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{D_n}})$ by $\mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{\boldsymbol{D}}}$.

Let $\mathcal{M}$ be the message space. Given a positive integer n, an *n-designated verifier signature* (n-DVS) scheme is defined by the following algorithms:

- Setup is a probabilistic algorithm that outputs the public parameter *param*;
- $\mathsf{KeyGen}(\mathit{param})$ is a probabilistic algorithm that takes the public parameters as an input and outputs a secret/public key-pair $(\mathsf{SK}, \mathsf{PK})$;
- $\mathsf{Sign}_{\mathsf{SK}_S,\mathsf{PK}_{\boldsymbol{D}}}(m)$ takes as inputs signer's secret key, designated verifiers' public keys, a message $m \in \mathcal{M}$ and a possible random string, and outputs a signature σ;
- For $i \in [1, n]$, $\mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{D_i}}(m)$ takes as inputs signer's public key, designated verifiers' public keys, secret key of one designated verifier, a message $m \in \mathcal{M}$ and a possible random string, and outputs a signature σ;
- $\mathsf{Verify}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}}}(m, \sigma)$ is a deterministic algorithm that takes as inputs a signing public key PK_S, public keys of all designated verifiers D_i, $i \in [1, n]$, a message $m \in \mathcal{M}$ and a candidate signature σ, and returns accept or reject;

If $n = 1$, we obtain a *designated verifier signature* (DVS) scheme. We say that a signature σ on m is valid if $\mathsf{Verify}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}}}(m, \sigma) = \mathsf{accept}$. As usually, we require that an n-DVS scheme is correct, that is, for all $(\mathsf{SK}_S, \mathsf{PK}_S)$ and $(\mathsf{SK}_{\boldsymbol{D}}, \mathsf{PK}_{\boldsymbol{D}})$ output by KeyGen,

for any $i \in [1,n]$ and for all $m \in \mathcal{M}$ we have $\mathsf{Verify}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}}}(\mathsf{Sign}_{\mathsf{SK}_S,\mathsf{PK}_{\boldsymbol{D}}}(m)) = \mathsf{Verify}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}}}(\mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{D_i}}(m)) = \mathsf{accept}$.

Let $\Delta = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Simul}, \mathsf{Verify})$ be an n-DVS scheme with the message space $\mathcal{M}$. Let Ω denote the space from which the random oracle H is selected; definition without a random oracle is analogous. Let $\mathcal{F}_m$ denote the adversary $\mathcal{F}$ with m as its input, and we assume that oracle calls are counted as one step.

It is required that a designated verifier signature satisfies the following three properties [LWB05]. We give the definitions of unforgeability and non-transferability only for the sake of completeness since we will not need them in this paper.

Unforgeability: Let $\mathcal{F}$ be an adversary against DVS. We define advantage $\mathsf{Adv}^{\mathsf{forge}}_{\Delta}(\mathcal{F})$ of $\mathcal{F}$ to be the next probability:

$$\Pr\left[\begin{array}{l} H \leftarrow \Omega; (\mathsf{SK}_S, \mathsf{PK}_S) \leftarrow \mathsf{KeyGen}; \\ (\mathsf{SK}_{D_1}, \mathsf{PK}_{D_1}) \leftarrow \mathsf{KeyGen}; \ldots; (\mathsf{SK}_{D_n}, \mathsf{PK}_{D_n}) \leftarrow \mathsf{KeyGen}; \\ (m,\sigma) \leftarrow \mathcal{F}^{\mathsf{Sign}_{\mathsf{SK}_S,\mathsf{PK}_{\boldsymbol{D}}}(\cdot),\mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{\boldsymbol{D}}}(\cdot),H(\cdot)}(\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}}) : \\ \sigma \notin \Sigma_S(m) \wedge \sigma \notin \Sigma_{D_1}(m) \wedge \cdots \wedge \sigma \notin \Sigma_{D_n}(m) \wedge \\ \mathsf{Verify}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}}}(m,\sigma) = \mathsf{accept} \end{array}\right],$$

where $\Sigma_S(m)$ is the set of signatures received from $\mathsf{Sign}_{\mathsf{SK}_S,\mathsf{PK}_{\boldsymbol{D}}}(m)$ and $\Sigma_{D_i}(m)$ is the set of signatures received from $\mathsf{Simul}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{D_i}}(m)$. $\mathcal{F}$ is said to $(\tau, q_h, q_s, \varepsilon)$-forge σ if $\mathcal{F}$ runs in time at most τ , makes at most q_h hash queries and in total at most q_s signing and simulation queries, and $\mathsf{Adv}^{\mathsf{forge}}_{\Delta}(\mathcal{F}) \geq \varepsilon$. A designated verifier signature scheme is $(\tau, q_h, q_s, \varepsilon)$-unforgeable if no forger can $(\tau, q_h, q_s, \varepsilon)$-forge it.

Non-transferability (informal): given a message-signature pair (m, σ) which is accepted by a designated verifier, and without access to the signer's secret key, it is computationally infeasible to determine whether the message was signed by the signer, or the signature was simulated by the designated verifier.

Non-delegatability: Let $\kappa \in [0,1]$ be the knowledge error. Δ is (τ, κ)-non-delegatable if there exists a black-box knowledge extractor $\mathcal{K}$ that, for every algorithm $\mathcal{F}$ and for every valid signature σ, satisfies the following condition: For every $(\mathsf{SK}_S, \mathsf{PK}_S) \leftarrow \mathsf{KeyGen}$, $(\mathsf{SK}_{D_i}, \mathsf{PK}_{D_i}) \leftarrow \mathsf{KeyGen}$, for $i \in [1,n]$, and message m, if $\mathcal{F}$ produces a valid signature on m with probability $\varepsilon > \kappa$, then on input m and on access to the oracle $\mathcal{F}_m$, $\mathcal{K}$ produces one of the secret keys $(\mathsf{SK}_S, \mathsf{SK}_{D_1}, \ldots, \mathsf{SK}_{D_n})$ in expected time $\frac{\tau}{\varepsilon - \kappa}$ (without counting the time to make the oracle queries). Here, $\mathcal{F}$'s probability is taken over the choice of her random coins and over the choice of $H \leftarrow \Omega$.

Variations of n-DVS. We call an n-DVS a *strong n-DVS* if the verification algorithm also takes an SK_{D_i}, $i \in [1,n]$, as an input, and verification without SK_{D_i}, for some $i \in [i,n]$, is computationally difficult. An n-DVS scheme is a designated multi verifier signature scheme if verification can be performed only by the coalition of all n designated verifiers. An n-DVS scheme is universal if it contains a conventional signing

algorithm (w.r.t. no designated verifier) and an arbitrary entity can convert the conventional signature to a signature w.r.t. an arbitrary designated verifier. An n-DVS scheme is ID-based if the public key of an arbitrary participant A can be computed from his or her ID ID_A.

3 Four DVS Schemes And Attacks on Them

3.1 SZM04 Scheme

Description. The SZM04 strong ID-based universal DVS scheme [SZM04] can be described as follows:

- Setup: A trusted authority (TA) generates two groups $(\mathbb{G}, +)$ and $(\mathbb{H}, \cdot)$ of prime order q and a bilinear mapping $\langle \cdot, \cdot \rangle : \mathbb{G} \times \mathbb{G} \to \mathbb{H}$, together with an arbitrary generator $P \in \mathbb{G}$. TA selects a master key $s \in \mathbb{Z}_q$ and set $P_{pub} \leftarrow sP$. Let $H_{\mathbb{G}} : \{0,1\}^* \to \mathbb{G}$ and $H_q : \{0,1\}^* \to \mathbb{Z}_q$ be two random oracles. The system parameters are $(q, \mathbb{G}, \mathbb{H}, \langle \cdot, \cdot \rangle, P, P_{pub}, H_{\mathbb{G}}, H_q)$.
- $\mathsf{KeyGen}(param)$: S and D publish their identities $\mathsf{PK}_S \leftarrow H_{\mathbb{G}}(\mathsf{ID}_S)$ and $\mathsf{PK}_D \leftarrow H_{\mathbb{G}}(\mathsf{ID}_D)$. Their secret keys are defined by TA as $\mathsf{SK}_S \leftarrow s \cdot \mathsf{PK}_S$ and $\mathsf{SK}_D \leftarrow s \cdot \mathsf{PK}_D$.
- $\mathsf{Sign}_{\mathsf{SK}_S, \mathsf{PK}_D}(m)$: to sign a message m for D, S generates two random values $k \leftarrow \mathbb{Z}_q$, $t \leftarrow \mathbb{Z}_q^*$, and computes $c \leftarrow \langle \mathsf{PK}_D, P \rangle^k$, $r \leftarrow H_q(m, c)$, $T \leftarrow t^{-1}kP - r \cdot \mathsf{SK}_S$. The signature is (T, r, t).
- $\mathsf{Simul}_{\mathsf{PK}_S, \mathsf{SK}_D}(m)$: To simulate the transcript on an arbitrary message m, D generates random $R \in \mathbb{G}$ and $a \in \mathbb{Z}_q^*$, and computes $c \leftarrow \langle R, \mathsf{PK}_D \rangle \cdot \langle \mathsf{PK}_S, \mathsf{SK}_D \rangle^a$, $r \leftarrow H_q(m, c)$, $t \leftarrow r^{-1}a \mod q$, $T \leftarrow t^{-1}R$. The transcript (T, r, t) is indistinguishable from the real signature [SZM04, Thm. 3].
- $\mathsf{Verify}_{\mathsf{PK}_S, \mathsf{SK}_D}(m, \sigma)$: given (m, T, r, t), D verifies its validity by testing whether $H_q(m, (\langle T, \mathsf{PK}_D \rangle \cdot \langle \mathsf{PK}_S, \mathsf{SK}_D \rangle^r)^t) = r$.

First attack. For both simulation and verification, it is sufficient to know $\langle \mathsf{SK}_S, \mathsf{PK}_D \rangle = \langle \mathsf{PK}_S, \mathsf{SK}_D \rangle$. Therefore, either the signer or the verifier can delegate the signing rights of S w.r.t. a fixed designated verifier D.

Second attack. Assume that the signer discloses $(k, k \cdot \mathsf{SK}_S)$ to any third party T, where $k \leftarrow \mathbb{Z}_q^*$. Given an arbitrary message $\tilde{m}$ and an arbitrary designated verifier D, T chooses random values $R \leftarrow \mathbb{G}$, $a \leftarrow \mathbb{Z}_q^*$ and computes $\tilde{c} \leftarrow \langle R, \mathsf{PK}_D \rangle \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D \rangle^{a(k^{-1}+1)}$, $\tilde{r} \leftarrow H_q(\tilde{m}, \tilde{c})$, $\tilde{t} \leftarrow \tilde{r}^{-1}a \mod q$, $\tilde{T} \leftarrow \tilde{t}^{-1}R + \tilde{r}k \cdot \mathsf{SK}_S$, obtaining a simulated signature $(\tilde{T}, \tilde{r}, \tilde{t})$. D can verify whether $H_q(\tilde{m}, (\langle \tilde{T}, \mathsf{PK}_D \rangle \cdot \langle \mathsf{PK}_S, \mathsf{SK}_D \rangle^{\tilde{r}})^{\tilde{t}}) = \tilde{r}$. The verification accepts since $\langle \tilde{T}, \mathsf{PK}_D \rangle^{\tilde{t}} = \langle \tilde{t}^{-1}R + \tilde{r}k \cdot$

$\mathsf{SK}_S, \mathsf{PK}_D\rangle^{\tilde{t}} = \langle R, \mathsf{PK}_D\rangle \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D\rangle^{\tilde{r}\tilde{t}} = \langle R, \mathsf{PK}_D\rangle \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D\rangle^{a}$ and

$$\begin{aligned}(\langle \tilde{T}, \mathsf{PK}_D\rangle \cdot \langle \mathsf{PK}_S, \mathsf{SK}_D\rangle^{\tilde{r}})^{\tilde{t}} &= \langle \tilde{T}, \mathsf{PK}_D\rangle^{\tilde{t}} \cdot \langle \mathsf{PK}_S, \mathsf{SK}_D\rangle^{\tilde{r}\tilde{t}} \\ &= \langle R, \mathsf{PK}_D\rangle \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D\rangle^{a} \cdot \langle \mathsf{SK}_S, \mathsf{PK}_D\rangle^{\tilde{r}\tilde{t}} \\ &= \langle R, \mathsf{PK}_D\rangle \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D\rangle^{a} \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D\rangle^{ak^{-1}} \\ &= \langle R, \mathsf{PK}_D\rangle \cdot \langle k \cdot \mathsf{SK}_S, \mathsf{PK}_D\rangle^{a(k^{-1}+1)} = \tilde{c} \ .\end{aligned}$$

Therefore, $H_q(\tilde{m}, (\langle \tilde{T}, \mathsf{PK}_D\rangle \cdot \langle \mathsf{PK}_S, \mathsf{SK}_D\rangle^{\tilde{r}})^{\tilde{t}}) = H_q(\tilde{m}, \tilde{c}) = \tilde{r}$. Thus, according to this attack, an arbitrary party who knows $(k, k \cdot \mathsf{SK}_S)$, for some k, can simulate signer's signature w.r.t. all designated verifiers.

3.2 NSM05 Scheme

The Ng-Susilo-Mu [NSM05] universal designated multi-verifier signature (UDMVS) scheme is as follows (here, $\mathcal{M} = \mathbb{Z}_q^*$):

- Setup: Choose a group pair $(\mathbb{G}, \mathbb{H})$ of prime order $|\mathbb{G}| = |\mathbb{H}| = q$, a bilinear map $\langle \cdot, \cdot \rangle : \mathbb{G} \times \mathbb{G} \to \mathbb{H}$, an arbitrary generator $P \in \mathbb{G}$ and a cryptographic hash function $H_{\mathbb{G}} : \{0,1\}^* \to \mathbb{G}$. The common parameter is $param = (q, \mathbb{G}, \mathbb{H}, \langle \cdot, \cdot \rangle, P, H_{\mathbb{G}})$.
- $\mathsf{KeyGen}(param)$: Given the common parameter, a participant picks a random $\mathsf{SK} \leftarrow \mathbb{Z}_q^*$, and computes $\mathsf{PK} \leftarrow \mathsf{SK} \cdot P$. The public key is PK and the secret key is SK. Thus, S has key pair $(\mathsf{SK}_S, \mathsf{PK}_S)$ and D has key pair $(\mathsf{SK}_D, \mathsf{PK}_D)$.
- $\mathsf{Sign}_{\mathsf{SK}_S, \mathsf{PK}_D}(m)$: Compute $\hat{\sigma} \leftarrow \mathsf{SK}_S \cdot H_{\mathbb{G}}(m)$, $\sigma \leftarrow \langle \hat{\sigma}, \sum_{i=1}^{n} \mathsf{PK}_{D_i} \rangle$. Return σ.
- $\mathsf{Verify}_{\mathsf{PK}_S, \mathsf{PK}_D, \mathsf{SK}_D}(m, \sigma)$: Each verifier D_i does the following: compute $\tilde{\sigma}_i \leftarrow \mathsf{SK}_{D_i} \cdot H_{\mathbb{G}}(m)$ and send it to other $n-1$ verifiers. After receiving all $\tilde{\sigma}_j$, $j \neq i$, validate all $\tilde{\sigma}_j$ by verifying that $\langle P, \tilde{\sigma}_j \rangle = \langle \mathsf{PK}_j, H_{\mathbb{G}}(m) \rangle$ for $j \neq i, j \in [1, n]$. Return reject if any of the verifications fails. Return accept if $\sigma = \prod_{i=1}^{n} \langle \tilde{\sigma}_i, \mathsf{PK}_S \rangle$, or reject otherwise.

Attack on NSM05. Denote $P_{sum} := \sum_{i=1}^{n} \mathsf{PK}_{D_i}$. If signer leaks $\mathsf{SK}_S \cdot P_{sum}$ to T, then T can compute

$$\sigma \leftarrow \langle H_{\mathbb{G}}(m), \mathsf{SK}_S \cdot P_{sum} \rangle = \langle \mathsf{SK}_S \cdot H_{\mathbb{G}}(m), P_{sum} \rangle = \langle \hat{\sigma}, P_{sum} \rangle \ .$$

After receiving (m, σ), each verifier i computes $\tilde{\sigma}_i \leftarrow \mathsf{SK}_{D_i} \cdot H_{\mathbb{G}}(m)$, and verifies that $\langle P, \tilde{\sigma}_j \rangle = \langle \mathsf{PK}_j, H_{\mathbb{G}}(m) \rangle$ for $j \neq i, j \in [1, n]$. Now, $\sigma = \prod_{i=1}^{n} \langle \tilde{\sigma}_i, \mathsf{PK}_S \rangle$ since

$$\begin{aligned}\sigma &= \langle H_{\mathbb{G}}(m), \mathsf{SK}_S \cdot P_{sum} \rangle = \langle \mathsf{SK}_S \cdot H_{\mathbb{G}}(m), P_{sum} \rangle = \langle \hat{\sigma}, P_{sum} \rangle \\ &= \prod_{i=1}^{n} \langle \hat{\sigma}, \mathsf{SK}_{D_i} \cdot P \rangle = \prod_{i=1}^{n} \langle \mathsf{SK}_S \cdot H_{\mathbb{G}}(m), \mathsf{SK}_{D_i} \cdot P \rangle \\ &= \prod_{i=1}^{n} \langle \mathsf{SK}_{D_i} \cdot H_{\mathbb{G}}(m), \mathsf{SK}_S \cdot P \rangle \\ &= \prod_{i=1}^{n} \langle \tilde{\sigma}_i, \mathsf{PK}_S \rangle \ .\end{aligned}$$

Therefore, for any message m, T can simulate signer's signature σ and pass the verification equation. Note that alternatively, all verifiers can cooperate by leaking $\sum \mathsf{SK}_{D_i} \cdot \mathsf{PK}_S = \mathsf{SK}_S \cdot P_{sum}$. Therefore, the NSM05 scheme is delegatable.

Additionally, Ng, Susilo and Mu first proposed a "simple" UDMVS scheme that is based on the universal DVS from [SBWP03]. There, analogously, if signer leaks $\mathsf{SK}_S \cdot \mathsf{PK}_{D_i}$ to T, then T can compute $\sigma_i = \langle \mathsf{SK}_S \cdot \mathsf{PK}_{D_i}, H_{\mathbb{G}}(m) \rangle = \langle \mathsf{PK}_{D_i}, \hat{\sigma} \rangle$, for $i \in [1, n]$. Verifier will accept σ_i for that $\sigma_i = \langle \mathsf{SK}_S \cdot \mathsf{PK}_{D_i}, H_{\mathbb{G}}(m) \rangle = \langle \mathsf{SK}_S \mathsf{SK}_{D_i} P, H_{\mathbb{G}}(m) \rangle = \langle \mathsf{SK}_S \cdot P, H_{\mathbb{G}}(m) \rangle^{\mathsf{SK}_{D_i}} = \langle \mathsf{PK}_S, H_{\mathbb{G}}(m) \rangle^{\mathsf{SK}_{D_i}}$ for $i \in [1, n]$.

Furthermore, our attack works also with the MDVS scheme from [NSM05] because its signing algorithm is same as the signing algorithm in the UDMVS scheme.

3.3 ZFI05 Scheme

The next strong DVS scheme ZFI05 was proposed in [ZFI05] (we describe a slightly simplified version of ZFI05, but our attack works also with the original version):

- Setup: Choose a bilinear group pair $(\mathbb{G}, \mathbb{H})$ of prime order $|\mathbb{G}| = |\mathbb{H}| = q$, with a bilinear map $\langle \cdot, \cdot \rangle : \mathbb{G} \times \mathbb{H} \to \mathbb{H}$ and an isomorphism $\psi : \mathbb{H} \to \mathbb{G}$. Here, $\mathbb{G}$ is multiplicative. Choose a random generator $g_2 \in \mathbb{H}$, and compute $g_1 = \psi(g_2) \in \mathbb{G}$. Then the common parameter is $param = (q, \mathbb{G}, \mathbb{H}, \langle \cdot, \cdot \rangle, \psi, g_1, g_2)$.
- $\mathsf{KeyGen}(param)$: Pick random $x, y \leftarrow \mathbb{Z}_q^*$, compute $u \leftarrow g_2^x$, $v \leftarrow g_2^y$. The public key is $\mathsf{PK} \leftarrow (u, v)$ and the secret key is $\mathsf{SK} \leftarrow (x, y)$. In particular, S has a key pair with $\mathsf{PK}_S = (u_S, v_S)$, $\mathsf{SK}_S = (x_S, y_S)$ and D has a key pair with $\mathsf{PK}_D = (u_D, v_D)$, $\mathsf{SK}_D = (x_D, y_D)$.
- $\mathsf{Sign}_{\mathsf{SK}_S, \mathsf{PK}_D}(m)$: Pick a random $r \leftarrow \mathbb{Z}_q^*$. If $x_S + r + y_S m \equiv 0 \mod q$, restart. Compute $\sigma' \leftarrow g_1^{1/(x_S + r + y_S m)} \in \mathbb{G}$, $h \leftarrow g_2^r$, $d \leftarrow \langle u_D, v_D^r \rangle \in \mathbb{H}$. Return $\sigma \leftarrow (\sigma', h, d)$.
- $\mathsf{Simul}_{\mathsf{PK}_S, \mathsf{SK}_D}(m)$: Pick a random $s \in \mathbb{Z}_q^*$ and compute $\sigma' \leftarrow g_2^s$, $h \leftarrow g_2^{1/s} u_S^{-1} v_S^{-m}$ and $d \leftarrow \langle g_1, h \rangle^{x_D y_D}$. Return $\sigma \leftarrow (\sigma', h, d)$.
- $\mathsf{Verify}_{\mathsf{PK}_S, \mathsf{SK}_D}(\sigma', h, d)$: Output accept if $\langle g_1, g_2 \rangle = \langle \sigma', u_S \cdot h \cdot v_S^m \rangle$ and $d = \langle u_D, h^{y_D} \rangle$. Otherwise, output reject.

Attack on ZFI05. In simulation algorithm, the designated verifier can compute d as $d \leftarrow \langle g_1^{x_D y_D}, h \rangle$. Thus, designated verifier can reveal $g_1^{x_D y_D}$, and therefore this scheme is delegatable by the verifier. Note that the delegation token does not depend on the signer.

3.4 LV04 Scheme

Description. In [LV04b], Laguillaumie and Vergnaud proposed the next 2-DVS scheme based on bilinear maps. Here, D_1 and D_2 are two verifiers specified by signer S. $\mathbb{G}$ and $\mathbb{H}$ are groups of order q, P generates $\mathbb{G}$, and $\langle \cdot, \cdot \rangle : \mathbb{G} \times \mathbb{G} \to \mathbb{H}$ is an admissible bilinear map. Let $BDHGen$ be a Bilinear Diffie-Hellman (BDH)-prime order generator [LV04b].

– Setup: Set $(q, \mathbb{G}, \mathbb{H}, \langle\cdot,\cdot\rangle, P) \leftarrow BDHGen$, and let $H_{\mathbb{G}}$ be a random member of a hash function family from $\{0,1\}^* \times \mathbb{H} \to \mathbb{G}$. The common parameter is $(q, \mathbb{G}, \mathbb{H}, \langle\cdot,\cdot\rangle, P, H_{\mathbb{G}})$.
– $\mathsf{KeyGen}(param)$: Pick a random $\mathsf{SK} \leftarrow \mathbb{Z}_q^*$, and compute $\mathsf{PK} \leftarrow \mathsf{SK}\cdot P$. The public key is PK and the secret key is SK.
– $\mathsf{Sign}_{\mathsf{SK}_S,\mathsf{PK}_{D_1},\mathsf{PK}_{D_2}}(m)$: Given a message $m \in \{0,1\}^*$, S picks at random two integers $(r,\ell) \in \mathbb{Z}_q^* \times \mathbb{Z}_q^*$, computes $u \leftarrow \langle \mathsf{PK}_{D_1}, \mathsf{PK}_{D_2}\rangle^{\mathsf{SK}_S}$, $Q_1 \leftarrow {\mathsf{SK}_S}^{-1}(H_{\mathbb{G}}(m,u^\ell) - r(\mathsf{PK}_{D_1} + \mathsf{PK}_{D_2}))$ and $Q_2 \leftarrow rP$. The signature is $\sigma = (Q_1, Q_2, \ell)$.
– $\mathsf{Verify}_{\mathsf{PK}_S,\mathsf{PK}_{\boldsymbol{D}},\mathsf{SK}_{D_i}}(m, Q_1, Q_2, \ell)$: Designated verifier D_i, where $i \in \{1,2\}$, computes $u \leftarrow \langle \mathsf{PK}_S, \mathsf{PK}_{D_{3-i}}\rangle^{\mathsf{SK}_{D_i}}$. He or she tests whether $\langle Q_1, \mathsf{PK}_S\rangle \cdot \langle Q_2, \mathsf{PK}_{D_1} + \mathsf{PK}_{D_2}\rangle = \langle H_{\mathbb{G}}(m,u^\ell), P\rangle$.

Attack on LV04. Suppose D_1 and D_2 collude to leak $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$ to T. Then T picks two random integers $\tilde{r}, \tilde{\ell} \leftarrow \mathbb{Z}_q^*$, computes $\tilde{M} \leftarrow H_{\mathbb{G}}(m, \tilde{\ell})$, $\tilde{Q}_1 \leftarrow \tilde{r}P$, and $\tilde{Q}_2 \leftarrow (\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2})^{-1}(\tilde{M} - \tilde{r}\cdot \mathsf{PK}_S)$. The simulated signature is $\tilde{\sigma} \leftarrow (\tilde{Q}_1, \tilde{Q}_2, \tilde{\ell})$. Verification accepts since

$$
\begin{aligned}
&\langle \tilde{Q}_1, \mathsf{PK}_S\rangle \cdot \langle \tilde{Q}_2, \mathsf{PK}_{D_1} + \mathsf{PK}_{D_2}\rangle \\
&\quad = \langle \tilde{r}P, \mathsf{PK}_S\rangle \cdot \langle (\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2})^{-1}(\tilde{M} - \tilde{r}\cdot \mathsf{PK}_S), \mathsf{SK}_{D_1}P + \mathsf{SK}_{D_2}\cdot P\rangle \\
&\quad = \langle \tilde{r}P, \mathsf{PK}_S\rangle \cdot \langle (\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2})^{-1}(\tilde{M} - \tilde{r}\cdot \mathsf{PK}_S), P\rangle^{\mathsf{SK}_{D_1}+\mathsf{SK}_{D_2}} \\
&\quad = \langle \tilde{r}P, \mathsf{PK}_S\rangle \cdot \langle \tilde{M} - \tilde{r}\cdot \mathsf{PK}_S, P\rangle \\
&\quad = \langle \tilde{M}, P\rangle \cdot \langle \tilde{r}\cdot \mathsf{PK}_S, P\rangle \cdot \langle -\tilde{r}\cdot \mathsf{PK}_S, P\rangle = \langle \tilde{M}, P\rangle \ .
\end{aligned}
$$

Therefore, if D_1 and D_2 collaborate then they can leak $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$ to T. After that, T will be able to simulate signatures of *any* signer w.r.t. the designated verifier pair (D_1, D_2).

Discussion. Our attack is based on the following observation: by the verification equation $\langle Q_1, \mathsf{PK}_S\rangle \cdot \langle Q_2, \mathsf{PK}_{D_1} + P_{D_2}\rangle = \langle H_{\mathbb{G}}(m,u^\ell), P\rangle$, we have $\langle Q_1, P\rangle^{\mathsf{SK}_S} \cdot \langle Q_2, P\rangle^{\mathsf{SK}_{D_1}+\mathsf{SK}_{D_2}} = \langle H_{\mathbb{G}}(m,u^\ell), P\rangle$. Here, attacker can choose only Q_1 and Q_2, and it must hold that $\mathsf{SK}_S \cdot Q_1 + (\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2})Q_2 = H_{\mathbb{G}}(m,u^\ell)$. Due to the random oracle assumption, $H_{\mathbb{G}}(m,u^\ell)$ is a random value, and thus either Q_1 or Q_2 must depend on $H_{\mathbb{G}}(m,u^\ell)$. This means that attacker either must know the value SK_S (and thus he can recover S's secret key), or the value $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$. Leaking SK_S is not an attack according to [LWB05], but leaking $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$ is.

To guarantee the non-transferability, the designated verifier(s) should have the capability to simulate correct signature transcripts. However, the LV04 scheme does not include simulation algorithms. The above attack can also be treated as two-party simulation algorithm if D_1 and D_2 execute it themselves.

Although a third party cannot deduce SK_{D_1} or SK_{D_2} from $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$, we need that two parties D_1 and D_2 compute $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$ together. This means that either

these two parties must trust each other, or they have to execute a secure two-party computation.

Finally, note that both this attack and the attack against ZFI05 have the same feature: if the delegation token revealed by verifier(s) (either $\mathsf{SK}_{D_1} + \mathsf{SK}_{D_2}$ or $g_1^{x_{D_1} y_{D_1}}$) is not connected anyhow to the signer. Therefore, after revealing those values, a third party can simulate the signature of *any* signer w.r.t. a fixed designated verifier or a fixed pair of designated verifiers.

4 On Different Delegation Attacks

Who can delegate? In the previous section, we saw at least two kinds of delegation attacks:

Attack I: Either the signer or one of the designated verifiers can delegate the signing rights to a third party T without disclosing his or her secret key.

Attack II: One of the designated verifiers (or even only the coalition of all verifiers) can delegate the signing right to a third party without disclosing his or her secret key, while the signer cannot do it.

The non-delegatability notion introduced in [LWB05] corresponds to security against Attack I. Next, we will give a somewhat formal definition of what we mean by a vulnerability to Attack II. (Intuitively, it says that a n-DVS scheme Δ is verifier-only delegatable if it is delegatable but it cannot be delegated by the signer without leaking Signer's secret key.)

Verifier-only delegatability: As previously, $\mathcal{F}_m$ denotes $\mathcal{F}$ with m as its input, and oracle calls are counted as one step. More precisely, let $\kappa \in [0,1]$ be the knowledge error. We say that Δ is verifier-only (τ,κ)-delegatable if it is not (τ,κ)-non-delegatable and there exists a black-box knowledge extractor $\mathcal{K}$ that, for every algorithm $\mathcal{F}$ and for every message $m \in \mathcal{M}$ satisfies the following condition: for every $(\mathsf{SK}_S, \mathsf{PK}_S) \leftarrow \mathsf{KeyGen}$, $(\mathsf{SK}_{D_1}, \mathsf{PK}_{D_1}) \leftarrow \mathsf{KeyGen}$, ..., $(\mathsf{SK}_{D_n}, \mathsf{PK}_{D_n}) \leftarrow \mathsf{KeyGen}$, for every bit-string d (delegation token) that does not depend on SK_{D_i} for any $i \in [1,n]$, and for any message m, if $\mathcal{F}$ produces a valid signature on m with probability $\varepsilon > \kappa$ then, on input m and on access to the oracle $\mathcal{F}_m$, $\mathcal{K}$ produces SK_{D_i} for some $i \in [1,n]$ in expected time $\frac{\tau}{\varepsilon - \kappa}$ (without counting the time to make the oracle queries). Here again, $\mathcal{F}$'s probability is taken over the choice of her random coins and over the choice of the random oracles.

What exactly can be delegated? The presented attacks can be divided into delegation attacks that allow to delegate the signing rights of a fixed signer w.r.t. a fixed tuple of designated verifiers, or the rights of *any* signer w.r.t. a fixed tuple of designated verifiers, or the rights of a fixed signer w.r.t. any tuple of designated verifiers. According to [LWB05], existence of any of these attacks makes a scheme delegatable. Again, it can be argued that the first type of attack is the most serious one since the last two attack types give too much power to the third party T (and therefore one might be less

motivated to delegate the rights). One can try to modify the delegatability definition so that only the first attack type is classified as an attack. We will leave it as an open question whether this is reasonable.

Final remarks. Regardless of the previous comments, our own opinion is that our attacks against all four schemes indicate some weaknesses in them and while the non-delegatability definition of [LWB05] might be too strong in some sense, to avoid any kind of future attack and unexpected vulnerabilities, it is a good idea to design DVS schemes that are non-delegatable according to [LWB05].

Acknowledgements

The first and the third authors are partially supported by the National Grand Fundamental Research 973 Program of China under Grant No. G1999035804. The second author is partially supported by the Estonian Science Foundation, grant 6096. The authors would like to thank Guilin Wang and the anonymous referees for useful comments.

References

[Cha96] David Chaum. Private Signature and Proof Systems, 1996. US-patent no 5,493,614.

[JSI96] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated Verifier Proofs and Their Applications. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 143–154, Saragossa, Spain, May 12–16, 1996. Springer-Verlag.

[LV04a] Fabien Laguillaumie and Damien Vergnaud. Designated Verifier Signatures: Anonymity and Efficient Construction from Any Bilinear Map. In Carlo Blundo and Stelvio Cimato, editors, *Security in Communication Networks, 4th International Conference, SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 105–119, Amalfi, Italy, September 8–10, 2004. Springer Verlag.

[LV04b] Fabien Laguillaumie and Damien Vergnaud. Multi-designated Verifiers Signatures. In Javier Lopez, Sihan Qing, and Eiji Okamoto, editors, *Information and Communications Security, 6th International Conference, ICICS 2004*, volume 3269 of *Lecture Notes in Computer Science*, pages 495–507, Malaga, Spain, October 27–29, 2004. Springer-Verlag.

[LWB05] Helger Lipmaa, Guilin Wang, and Feng Bao. Designated Verifier Signature Schemes: Attacks, New Security Notions and A New Construction. In Luis Caires, Guiseppe F. Italiano, Luis Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *The 32nd International Colloquium on Automata, Languages and Programming, ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*, pages 459–471, Lisboa, Portugal, 2005. Springer-Verlag.

[NSM05] Ching Yu Ng, Willy Susilo, and Yi Mu. Universal Designated Multi Verifier Signature Schemes. In Cheng-Zhong Xu and Laurence T. Yang, editors, *The International Workshop on Security in Networks and Distributed Systems (SNDS 2005)*, Fukuoka, Japan, July 20–22, 2005. IEEE Press. To appear.

[SBWP03] Ron Steinfeld, Laurence Bull, Huaxiong Wang, and Josef Pieprzyk. Universal Designated-Verifier Signatures. In Chi Sung Laih, editor, *Advances on Cryptology — ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 523–542, Taipei, Taiwan, November 30–December 4, 2003. Springer-Verlag.

[SKM03] Shahrokh Saeednia, Steve Kremer, and Olivier Markowitch. An Efficient Strong Designated Verifier Signature Scheme. In Jong In Lim and Dong Hoon Lee, editors, *Information Security and Cryptology - ICISC 2003*, volume 2971 of *Lecture Notes in Computer Science*, pages 40–54, Seoul, Korea, November 27–28, 2003. Springer-Verlag.

[SWP04] Ron Steinfeld, Huaxiong Wang, and Josef Pieprzyk. Efficient Extension of Standard Schnorr/RSA Signatures into Universal Designated-Verifier Signatures. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, *Public Key Cryptography 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 86–100, Singapore, March 1–4, 2004. Springer-Verlag.

[SZM04] Willy Susilo, Fangguo Zhang, and Yi Mu. Identity-Based Strong Designated Verifier Signature Schemes. In Josef Pieprzyk and Huaxiong Wang, editors, *The 9th Australasian Conference on Information Security and Privacy (ACISP 2004)*, volume 3108 of *Lecture Notes in Computer Science*, pages 313–324, Sydney, Australia, July 13–15, 2004. Springer-Verlag.

[ZFI05] Rui Zhang, Jun Furukawa, and Hideki Imai. Short Signature and Universal Designated Verifier Signature Without Random Oracles. In John Ioannidis, Angelos D. Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security, Third International Conference, ACNS 2005*, volume 3531 of *Lecture Notes in Computer Science*, pages 483–498, New York, NY, USA, June 7–10, 2005. Springer-Verlag.