PROXIABLE DESIGNATED VERIFIER SIGNATURE

Mebae Ushida, Kazuo Ohta
Graduate School of Information and Communication Engineering, University of Electro Communication, Chofu, Tokyo, Japan

Yutaka Kawai
Department of Mathematical Engineering and Information Physics, Faculty of Engineering
The University of Tokyo, Chiba, Japan

Kazuki Yoneyama
NTT Information Sharing Platform Laboratories, Tokyo, Japan

Keywords: Designated verifier signature, Proxy, Strong unforgeability.

Abstract: Designated Verifier Signature (DVS) guarantees that only a verifier designated by a signer can verify the *"validity of a signature"*. In this paper, we propose a new variant of DVS; Proxiable Designated Verifier Signature (PDVS) where the verifier can make a third party (i.e. the proxy) substitute some process of the verification. In the PDVS system, the verifier can reduce his computational cost by delegating some process of the verification without revealing the validity of the signature to the proxy. In all DVS systems, the validity of a signature means that a signature satisfies both properties that (1) the signature is judged *"accept"* by a decision algorithm and (2) the signature is confirmed at it is generated by the signer. So in the PDVS system, the verifier can make the proxy substitute checking only the property of (1). In the proposed PDVS model, we divide verifier's secret keys into two parts; one is a key for performing the decision algorithm, and the other is a key for generating a dummy signature, which prevents a third party from convincing the property (2). We also define security requirements for the PDVS, and propose a PDVS scheme which satisfies all security requirements we define.

1 INTRODUCTION

1.1 Background

Designated Verifier Signature (DVS) was first introduced by Jakobsson, Sako and Impagliazzo (Jakobsson et al., 1996). In the DVS system, a signer designates a verifier and only the verifier designated by the signer can verify the validity of a signature.

DVS is useful for a situation where a signer expects that the validity of the signature is confirmed by only specific person and is not confirmed by the others.

We consider the situation of public procedures. The person sends his personal information (a report of one's removal etc.) to the government office. And he hopes that this information cannot be leaked to others. He must generate his signature for this document, but he worries about leaking and being confirmed his personal information. If he uses the DVS, he can inform his personal information to the government and not have to worry about leaking it.

Another kind of signature where the signer can restrict to verify the validity of the signature is the Undeniable Signature (US) (Chaum and van Antwerpen, 1990). In the US system, the verifier needs the interaction with the signer to perform the verification. The signer designates the verifier by selecting the person whom the signer interacts with for verification. The third party who does not interact with the signer can not confirm the validity of the signature, and the verifier cannot convince the third party of validity of the signature which the verifier verified before by revealing the records of verification process.

In the US system, the verifier must interact with the signer whenever he verifies the signature. On the other hand in the DVS system, the signer designates the verifier when he generates the signature, and the verifier can verify the validity of the signature at any time without interaction with the signer.

Ushida M., Ohta K., Kawai Y. and Yoneyama K. (2010).
PROXIABLE DESIGNATED VERIFIER SIGNATURE.
In *Proceedings of the International Conference on Security and Cryptography*, pages 344-353
DOI: 10.5220/0002979403440353

By using Message Authenticate Code (MAC), the prover can also designate the verifier. MAC is also verified the validity without interaction. However the prover and the verifier must share a common secret key before using MAC. In the DVS system, the signer can designate the verifier using only the verifier's public key.

In the DVS system, the validity of a signature is checked by following two procedures: Decision and Distinction. By Decision, the signature is checked whether it is *"accepted"* by the decision procedure. By Distinction, the signature is checked whether it is exactly generated by the signer. In this paper, we call a signature which is accepted by Decision *an acceptable signature*, and a signature which is acceptable signature and generated by the signer *a valid signature*. The meaning of verifying the validity of a signature is confirming that the signature is valid by performing Decision and Distinction.

In the DVS system, the verifier can also generate an acceptable signature. We call such an acceptable signature *a dummy signature*, while we call a signature generated by a signer *an original signature*. Only the original signature must be confirmed as the valid signature. Any third party should be unable to distinguish the original signature from dummy signatures. Even if a third party accepts a signature, he is unable to confirm that the signature is the original signature because it could be a dummy signature. Therefore, a third party is unable to verify the validity of the signature. On the other hand, the verifier can decide whether the signature is the original signature by using his own list of dummy signatures generated by himself. Hence, the verifier cannot convince a third party the validity of the signature.

In several DVS systems (Jakobsson et al., 1996; Rivest et al., 2001; Lipmaa et al., 2005; Shahandashti and Safavi-Naini, 2008), anyone can perform the Decision. However, a third party cannot confirm the validity of a signature because he can not perform Distinction. We call those DVS systems *ordinary DVS*. In the ordinary DVS system, a third party can narrow the signer to two candidates. On the other hand, *strong DVS* (Saeednia et al., 2004; Laguillaumie and Vergnaud, 2005; Steinfeld et al., 2003) in which only the verifier can perform the Decision was proposed. In the strong DVS system, a third party cannot even narrow two signer candidates.

1.2 A Motivating Problem

In a strong DVS system, all processes of the verification can be performed by only a verifier. If one person is designated by large numbers of signers, he must proceed large amount of the task of the verification procedure by himself.

This situation will often occur if the DVS system is applied to the situation of public procedures. In this case, a lot of people would send their documents with DVSs to one government office. Then, the officer must verify large amount of DVSs. Hence, the officer would like to entrust other organizations to some processes of verification.

1.3 Contribution

In order to reduce the computational cost for verification, we will propose Proxiable Designated Verifier Signature (PDVS) where the verifier can make a third party (i.e. the proxy) substitute some process of the verification. In previous DVS systems, if the third party can perform the Decision, but he cannot confirm the validity of a signature. Hence in the PDVS system, the Decision is delegated to the proxy and the verifier performs only the Distinction. If the verifier does not issue any dummy signature for message m, he verifies that (m, σ) is valid immediately when he is reported that (m, σ) is acceptable by the proxy. Hence the verifier can reduce his computational cost.

In previous strong DVS systems (Saeednia et al., 2004; Laguillaumie and Vergnaud, 2005; Steinfeld et al., 2003), there is only one kind of verifier's secret key which is used for performing the Decision algorithm and for generating dummy signatures. If the verifier gives his secret key in order to delegate the Decision, the proxy can also generate a dummy signature. In this case, the verifier cannot perform the Distinction. Thus in the previous strong DVS systems, the verifier cannot delegate the verification task to the proxy.

Hence in the PDVS system, there are two kinds of verifier's keys; one is a key for performing the Decision and the other is for generating dummy signatures. The verifier can delegate the Decision to the proxy by giving only the secret key for performing the Decision, and the verifier keeps the both of keys; a key for performing the Decision and a key for generating dummy signatures.

Unlike the previous DVS systems, there is the new entity proxy in the PDVS system. Hence we consider the requirements for each position, not only the verifier and the third party but also the proxy. We define security requirements for PDVS scheme by capturing following requirements. (1) The verifier can surely verify the validity of the signature at any time. (2) The proxy can perform the Decision, but cannot generate any acceptable signature. (3) The third party cannot perform even the Decision. We describe the definition

of security requirements in Sect 3.2.

In this paper, we formalize PDVS, and define security requirements for PDVS in Sect 3. We propose a concrete PDVS scheme and prove that our PDVS scheme satisfies security requirements we define in Sect 3.2.

1.4 Related Works

In 1996, DVS (Jakobsson et al., 1996) was firs introduced and is the first ordinary DVS. After that, strong DVS (Saeednia et al., 2004) was proposed, and several security requirements for DVS was defined (Saeednia et al., 2004; Laguillaumie and Vergnaud, 2005; Lipmaa et al., 2005).

At the same time, several variants of DVS was proposed. *multi-DVS* (Laguillaumie and Vergnaud, 2004) is the DVS where the signer can designate several verifiers in one signature, and the verifiers can verify the signature individually. *Universal DVS* (Steinfeld et al., 2003; Steinfeld et al., 2004; Baek et al., 2005; Shahandashti and Safavi-Naini, 2008) is a system that a basic digital signature can convert a designated verifier signature. *designated proxy signature* (DVPS) (Wang, 2005) is the DVS where the signer can delegate his signing capacity to the third party (i.e. the proxy).

In all of the DVS system which was proposed before, the verifier have to verify the validity of the signature himself.

2 PRELIMINARIES

We will provide several definitions which are building blocks of our PDVS scheme.

Definition 1 (Bilinear Map). *Let $(\mathbb{G},+)$, and $(\mathbb{H},\cdot)$ be two groups of the same prime order q. Let P be a generator of $\mathbb{G}$. A bilinear map is a mapping $e:\mathbb{G}\times\mathbb{G}\to\mathbb{H}$ satisfying the following properties:*

- *bilinear: $e(aQ,bR)=e(Q,R)^{ab}$, for all $(Q,R)\in\mathbb{G}^2$, and all $(a,b)\in\mathbb{Z}^2$;*
- *non-degeneration: $e(P,P)\neq 1$;*
- *computability: there exists an efficient algorithm to compute e;*

Definition 2 (Prime Order BDH Parameter Generator). *Prime-order-BDH-parameter-generator is a probabilistic algorithm that takes on input a security parameter k, and outputs a 5-tuple $(q,P,\mathbb{G},\mathbb{H},e)$ satisfying the following conditions:*

- *q is a prime with $2^{k-1}<q<2^k$;*
- *$\mathbb{G}$ and $\mathbb{H}$ are groups of order q;*
- *$e:\mathbb{G}\times\mathbb{G}\longrightarrow\mathbb{H}$ is a bilinear map;*

Definition 3 (Computational Diffie-Hellman Assumption). *Let* Gen *be a Prime-order-BDH-parameter-generator. Let $\mathcal{A}$ be an adversary that takes on input 5-tuple $(q,P,\mathbb{G},\mathbb{H},e)$ generated by* Gen *and $(X,Y)\in\mathbb{G}^2$, and returns an elements of $Z\in\mathbb{G}$. We consider the following random experiments, where k is a security parameter;*

Experiment $\mathsf{Exp}^{cdh}_{Gen,\mathcal{A}}(k)$

$(q,P,\mathbb{G},\mathbb{H},e)\overset{R}{\leftarrow}\mathsf{Gen}(k)$

$(x,y)\overset{R}{\leftarrow}\mathbb{Z}_q^{*2}, X:=xP, Y:=yP$

$Z\leftarrow\mathcal{A}(q,P,\mathbb{G},\mathbb{H},e,X,Y)$

Return 1 iff $Z=xyP$

We define the corresponding success probability of $\mathcal{A}$ via

$$\mathsf{Succ}^{cdh}_{Gen,\mathcal{A}}(k)=Pr[\mathsf{Exp}^{cdh}_{Gen,\mathcal{A}}(k)=1].$$

Let $t\in\mathbb{N}$. CDH is said to be (k,t,ε)-hard if no adversary $\mathcal{A}$ running in time t has $\mathsf{Succ}^{cdh}_{Gen,\mathcal{A}}(k)\geq\varepsilon$.

Definition 4 (Gap-Bilinear Diffie-Hellman Assumption). *Let* Gen *be a Prime-order-BDH-parameter-generator. Let $\mathcal{A}$ be an adversary that takes on input 5-tuple $(q,P,\mathbb{G},\mathbb{H},e)$ generated by* Gen *and $(X,Y,Z)\in\mathbb{G}^3$, and returns an elements of $h\in\mathbb{H}$. We consider the following random experiments, where k is a security parameter;*

Experiment $\mathsf{Exp}^{gbdh}_{Gen,\mathcal{A}}(k)$

$(q,P,\mathbb{G},\mathbb{H},e)\overset{R}{\leftarrow}\mathsf{Gen}(k)$

$(x,y,z)\overset{R}{\leftarrow}\mathbb{Z}_q^{*3}, X:=xP, Y:=yP, Z:=zP$

$h\leftarrow\mathcal{A}^{DBDH}(q,P,\mathbb{G},\mathbb{H},e,X,Y,Z)$

Return 1 iff $h=e(P,P)^{xyz}$

where $\mathcal{A}^{DBDH}$ denotes that the adversary $\mathcal{A}$ has access to a DBDH oracle. A DBDH oracle is an oracle that for input aP,bP,cP, and $e(P,P)^d$, decides whether $d=abc$ or not. We define the corresponding success probability of $\mathcal{A}$ via

$$\mathsf{Succ}^{gbdh}_{Gen,\mathcal{A}}(k)=Pr[\mathsf{Exp}^{gbdh}_{Gen,\mathcal{A}}(k)=1].$$

Let $t\in\mathbb{N}$. GBDH is said to be (k,t,ε)-hard if no adversary $\mathcal{A}$ running in time t has $\mathsf{Succ}^{gbdh}_{Gen,\mathcal{A}}(k)\geq\varepsilon$.

3 DEFINITIONS OF PROXIABLE DVS

In this section, we will propose the definition of the PDVS and will several security properties of the PDVS.

3.1 The Models of PDVS Scheme

A PDVS scheme consists of seven algorithms : Let k be a security parameter. Each definition is described as follows.

Common parameter generation (SetUp): A probabilistic algorithm, on input k, outputs the public parameters *params*.

Signer's key generation (SKeyGen): A probabilistic algorithm, on input *params*, outputs the public and secret signer's key *PKs* and *SKs*.

Verifier's key generation (VKeyGen): A probabilistic algorithm, on input *params*, outputs verifier's secret key *SKv* and *SKp*, and the verifier's public key *PKv*. *SKv* is kept by only the verifier. *SKp* is given to the proxy by the verifier.

Designated signing (DSign): A probabilistic algorithm, on input *params*, message m, signer's secret key *SKs* and signer's and verifier's public keys PKs, PKv, outputs a original signature σ.

Transcript simulation (TSim): A probabilistic algorithm, on input *params*, message m, verifier's secret key *SKv*, and signer's and verifier's public keys PKs, PKv, outputs a dummy signature σ'.

Designated verifying 1 (Decision): A deterministic algorithm, on input *params*, message m, a signature σ, public key's PKs, PKv and verifier's secret key *SKp*, outputs a verification decision, *accept* or *reject*.

Designated verifying 2 (Distinction): A deterministic algorithm, on input *params*, message m, an acceptable signature σ, PKs, PKv, verifier's secret key *SKv* and the list of dummy signatures which the verifier issued before, outputs a verification decision, *valid* or *invalid*.

3.2 Definitions of Security Properties of PDVS

In this section, we propose definitions of security requirements for the PDVS.

3.2.1 Strong Unforgeability

We point out that *Existential Unforgeability* (EUF) is not sufficient and *Strong Existential Unforgeability* (sEUF) must be satisfied for secure PDVS schemes.

In the PDVS system satisfying EUF but not satisfying sEUF, the proxy is also able to confirm the validity of the signature.

We consider a following strong-forgery-attack. The strong-forgery-attacker generates an acceptable message/signature pair (m, σ^*) from another acceptable message/signature pair (m, σ). Anyone can not distinguish whether (m, σ^*) is generated by formal procedures (DSign or TSim) or the strong-forgery-attack. Such an attacker could exist in the PDVS system satisfying just EUF, because EUF only guarantees that anyone is unable to generate an acceptable (m^*, σ^*) where m^* is different from any acceptable signed message m.

If such a strong-forgery-attacker exists, the following situation occurs. The verifier generates a dummy signature σ_{TSim} for a message m, and issues (m, σ_{TSim}). Then the strong-forgery-attacker can generate a forgery (m, σ^*_{TSim}) by using (m, σ_{TSim}). After that, the signer generates an original signature σ_{DSign} for the message m. In this case, even if the verifier can decide that (m, σ) is acceptable, he cannot confirm where σ is the original signature σ_{DSign} or the forgery σ^*_{TSim}. Then even the verifier is unable to confirm the validity of the signature by the Distinction. So the verifier is unable to issue any dummy signature to confirm the validity of the signature in any cases. In the above situation, the proxy is able to confirm the validity of the signature by performing the Decision, because the acceptable signature is surely the original signature. Hence, if the PDVS does not satisfy sEUF, the proxy is able to confirm the validity of the signature. So, the PDVS must satisfy sEUF.

The PDVS requires that not only an arbitrary third party but the proxy, who has verifier's secret key *SKp*, is not able to forge a signature.

Definition 5 (Strong Unforgeability). [1] *Let $\mathcal{A}$ be a strong-forgery against adaptive chosen message attack (sEUF-CMA)-adversary against PDVS, Σ_S be the original signing oracle, Σ_T be the dummy signing oracle, and Υ be the distinction oracle* [2] *. Let $\{(m_1, \sigma_1), \cdots, (m_{q_{\Sigma_S}}, \sigma_{q_{\Sigma_S}})\}$ be a set of message and signature pair which is given to $\mathcal{A}$ by oracle Σ_S,*

[1] In the basic digital signature, the security notion of strong unforgeability is proposed by (An et al., 2002). We define strong unforgeability for the PDVS by adapting strong unforgeability to the PDVS system.

[2] The Decision oracle does not need in this experiment, because the adversary who has *SKp* can execute the Decision by himself.

$\{(m'_1,\sigma'_1),\cdots,(m'_{q_{\Sigma_T}},\sigma'_{q_{\Sigma_T}})\}$ *be a set of message and signature pair which is given to* $\mathcal{A}$ *by oracle* Σ_T*. Let k be a security parameter. We consider the following random experiment:*

Experiment $\mathsf{Exp}^{seuf-cma}_{PDVS,\mathcal{A}}(k)$

$params \xleftarrow{R} \mathsf{Setup}(k)$

$(PKs,SKs) \xleftarrow{R} \mathsf{SKeyGen}(params)$

$(PKv,SKv,SKp) \xleftarrow{R} \mathsf{VKeyGen}(params)$

$(m^*,\sigma^*) \leftarrow \mathcal{A}^{\Sigma_S,\Sigma_T,\Upsilon}(params,PKs,PKv,SKp)$

$s.t.\ (m^*,\sigma^*)$

$\notin \{(m_1,\sigma_1),\cdots,(m_{q_{\Sigma_S}},\sigma_{q_{\Sigma_S}})\}$

$\cup\{(m'_1,\sigma'_1),\cdots,(m'_{q_{\Sigma_T}},\sigma'_{q_{\Sigma_T}})\}$

Return 1

if $\mathsf{Decision}(params,m^*,\sigma^*,PKs,PKv,SKp) =$

$= accept$

We define the success probability of the adversary $\mathcal{A}$ *by*

$$\mathsf{Succ}^{seuf-cma}_{PDVS,\mathcal{A}}(k) = \mathsf{Pr}[\mathsf{Exp}^{seuf-cma}_{PDVS,\mathcal{A}}(k)=1].$$

A PDVS scheme is said to be (k,τ,ε)*-sEUF-CMA secure, if no adversary* $\mathcal{A}$ *running in time* τ *has a* $\mathsf{Succ}^{seuf-cma}_{PDVS,\mathcal{A}}(k) \geq \varepsilon$.

3.2.2 Privacy of Signer's Identity

In the PDVS system, a third party who has only public keys must be unable to confirm whether a signature is acceptable or not. To capture this requirement, we define *Privacy of signer's identity* (PSI) that "there are two possible signers. An adversary sees a signature σ, he is not able to distinguish the signer who generates σ." This condition can be described as follows.

Definition 6 (Privacy of Signer's Identity). *Let* $\mathcal{A}$ *be a PSI-CMA-adversary against PDVS,* Σ_{S_0} *and* Σ_{S_1} *be original signing oracles,* Σ_T *be the dummy signing oracle,* Γ *be the Decision oracle, and* Υ *be the Distinction oracle. Let k be a security parameter. We consider the following random experiment for* $i \in \{0,1\}$*.*

Experiment $\mathsf{Exp}^{psi-cma-i}_{PDVS,\mathcal{A}}(k)$

$params \xleftarrow{R} \mathsf{Setup}(k)$

$(PKs0,SKs0) \xleftarrow{R} \mathsf{SKeyGen}(params)$

$(PKs1,SKs1) \xleftarrow{R} \mathsf{SKeyGen}(params)$

$(PKv,SKv,SKp) \xleftarrow{R} \mathsf{VKeyGen}(params)$

$m^* \leftarrow \mathcal{A}^{\Sigma_{S_0},\Sigma_{S_1},\Sigma_T,\Gamma,\Upsilon}(params,PKs0,PKs1,PKv)$

$\sigma^* \leftarrow \mathsf{DSign}(params,m^*,SKsi,PKv)$

Return i'

$\leftarrow \mathcal{A}^{\Sigma_S,\Sigma_T,\Gamma,\Upsilon}(params,m^*,\sigma^*,PKs0,PKs1,PKv)$

We define the advantage of the adversary $\mathcal{A}$ *by*

$\mathsf{Adv}^{psi-cma}_{PDVS,\mathcal{A}}(k) =$

$|\mathsf{Pr}[\mathsf{Exp}^{psi-cma-0}_{PDVS,\mathcal{A}}(k)=1] - \mathsf{Pr}[\mathsf{Exp}^{psi-cma-1}_{PDVS,\mathcal{A}}(k)=1]|$

A PDVS scheme is said to be (k,τ,ε)*-PSI-CMA secure, if no adversary* $\mathcal{A}$ *running in time* τ *has* $\mathsf{Adv}^{psi-cma}_{PDVS,\mathcal{A}}(k) \geq \varepsilon$.

3.2.3 Source Hiding

In the PDVS system, anyone except the verifier who has all secret keys must be unable to confirm whether a signature is valid signature or not in order to guarantee that the Distinction is able to be performed by only the verifier. In this paper, *Source Hiding* (SH) means "even if any adversary $\mathcal{A}$ has all secret and public keys, he can not distinguish the original signature from the dummy signature."

It is clear that if a PDVS scheme satisfies SH, $\mathcal{A}$ who has a part of secret keys can not distinguish the original signature from the dummy signature. Thus if a scheme satisfies SH, the proxy can not confirm the validity of the signature.

Definition 7 (Source Hiding). *Let* $\mathcal{A}$ *be an arbitrary completely source hiding (SH)-adversary against a PDVS scheme. Let k be a security parameter. We consider the following random experiment:*

Experiment $\mathsf{Exp}^{sh}_{PDVS,\mathcal{A}}(k)$

$params \xleftarrow{R} \mathsf{Setup}(k)$

$(PKs,SKs) \xleftarrow{R} \mathsf{SKeyGen}(params)$

$(PKv,SKv,SKp) \xleftarrow{R} \mathsf{VKeyGen}(params)$

$m^* \leftarrow \mathcal{A}(params,PKs,PKv,SKs,SKv,SKp)$

$r \leftarrow_R \{0,1\}$

if $r=1 : \sigma^* \leftarrow \mathsf{DSign}(params,m^*,SKs,PKs,PKv)$

otherwise : $\sigma^* \leftarrow \mathsf{TSim}(params,m^*,SKv,PKs,PKv)$

$r' \leftarrow \mathcal{A}(params,m^*,\sigma^*,PKs,PKv,SKs,SKv,SKp)$

Return 1 iff $r'=r$

We define the advantage of the adversary $\mathcal{A}$ *by*

$$\mathsf{Adv}^{sh}_{PDVS,\mathcal{A}}(k) = |\mathsf{Pr}[\mathsf{Exp}^{sh}_{PDVS,\mathcal{A}}(k)=1] - \tfrac{1}{2}|.$$

A PDVS scheme is said to be (k,τ,ε)*-SH-CMA secure, if no adversary* $\mathcal{A}$ *running time* τ *has* $\mathsf{Adv}^{sh}_{PDVS,\mathcal{A}}(k) \geq \varepsilon$.

3.2.4 Non-coincidental Property

For message m, if the probability that $\sigma_{DSign} = \sigma_{TSim}$ such that $\sigma_{DSign} \leftarrow \mathsf{DSign}(params,m^*,SKs,PKs,PKv)$ and $\sigma_{TSim} \leftarrow$

$\mathsf{TSim}(params, m^*, SKv, PKs, PKv)$ is non-negligible, the verifier cannot confirm the validity of the signature. Since he cannot confirm that (m, σ_{DSign}) is the original signature because he cannot distinguish (m, σ_{DSign}) from the dummy signature (m, σ_{TSim}) he issued before.

Hence, the PDVS must satisfy the property that the provability that the original signature is identical with the dummy signature is negligible. In this paper, we call this property *Non-coincidental Property* (NCP).

Definition 8 (Non-coincidental Property). *A PDVS scheme is said to be (k, ε)-NCP secure, if for any m,*

$$\begin{aligned}&\Pr[\sigma_{DSign} = \sigma_{TSim} | params \leftarrow \mathsf{SetUp}(k);\\ &(SKs, PKs) \leftarrow \mathsf{SKeyGen}(params);\\ &(PKv, SKv, SKp) \stackrel{R}{\leftarrow} \mathsf{VKeyGen}(params)\\ &\sigma_{DSign} \leftarrow \mathsf{DSign}(params, m^*, SKs, PKs, PKv);\\ &\sigma_{TSim} \leftarrow \mathsf{TSim}(params, m^*, SKv, PKs, PKv)]\\ &\leq \varepsilon\end{aligned}$$

4 OUR PROPOSED PDVS SCHEME

In this section, we propose a PDVS scheme satisfying all security requirements which we defined in Sect 3.2.

First, we propose a naive PDVS scheme. But the naive PDVS scheme does not satisfy sEUF. Next, we show a strong-forgery attack for the naive PDVS scheme. Finally, we propose a PDVS scheme which is improved from the naive PDVS scheme and satisfies sEUF and other security requirements.

4.1 Naive PDVS Scheme

4.1.1 Idea

We achieve the naive PDVS scheme by using the bi-DVS scheme proposed by Laguillaumie and Vergnaud (Laguillaumie and Vergnaud, 2004). In the Bi-DVS, a signer designates two verifiers in one signature. The bi-DVS system does not capture dummy signatures and the validity of the signature is confirmed by only checking the Decision. Two verifiers have their own secret key respectively and can execute the Decision by using only his secret key [3].

We find that the bi-DVS scheme has a property where a person who has both two verifiers' secret keys can generate an acceptable signature without using signer's secret keys, and such acceptable signature is not distinguished from the signature generated by the signer. That is he can generate a dummy signature. We achieve the PDVS scheme by corresponding a key for performing the Decision to one of two verifiers' keys in the bi-DVS and keys for generating dummy signatures to both of two verifiers' keys.

4.1.2 Naive PDVS Scheme

Let k be a security parameter.

SetUp: Let Gen be a prime-order-BDH-generator and let $(q, P, \mathbb{G}, \mathbb{H}, e)$ be an output of Gen(k). Let $\mathcal{H} : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{H}$ be a hash function family and H be a random member of $\mathcal{H}$.

SKeyGen : Pick $a \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and compute $P_A = aP$. The signer's public key PKs is P_A and the secret key SKs is a.

VKeyGen: Pick $b \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and compute $P_B = bP$. Pick $c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and compute $P_c = cP$. The verifiers' public key PKv is P_B and P_C. The secret keys SKv which the verifier keeps are b and c, and the secret key SKp which the proxy is given by the verifier is c.

DSign : Given a message $m \in \{0,1\}^*$, pick $(r, l) \stackrel{R}{\leftarrow} \mathbb{Z}_q^{*2}$, compute $P_{BC} = P_B + P_C$, $u = e(P_B, P_C)^a$ and $M = H(m, u^l)$ and set $Q_A = a^{-1}(M - rP_{BC})$ and $Q_{BC} = rP$. The signature σ of m is (Q_A, Q_{BC}, l).

TSim: Given a message $m \in \{0,1\}^*$, pick $(r', l') \stackrel{R}{\leftarrow} \mathbb{Z}_q^{*2}$. Compute $P_{BC} = P_B + P_C$, $u = e(P_A, P_C)^b$ and $M' = H(m, u^{l'})$, and set $Q'_A = r'P$ and $Q'_{BC} = (b + c)^{-1}(M' - r'P_A)$. The dummy signature σ' of m is (Q'_A, Q'_{BC}, l').

Decision : Given m and σ, compute $u = e(P_A, P_B)^c$ and $M = H(m, u^l)$. Finally, check whether $e(Q_A, P_A)e(Q_{BC}, P_{BC}) = e(M, P)$. If it does, return $accept$. Otherwise return $reject$.

Distinction : Given an acceptable message/signature pair (m, σ), check whether $(m = m') \wedge (\sigma = \sigma')$ for any message/dummy signature pair (m', σ') which was issued before. If it does not, return $valid$. Otherwise return $invalid$.

[3] If each of verifiers can generate a dummy signature, the other verifier cannot confirm the validity of the signature. Because if it is so, there are more than two entities who can generate an acceptable signature and the verifier cannot confirm that the signature is generated by the signer. In the bi-DVS system, the validity of the signature is confirmed by only checking the Decision. So, each of verifiers can transfer the validity of the signature to a third party. Therefore, to be exact, the bi-DVS is not DVS.

4.1.3 Strong-forgery-attack for Naive PDVS Scheme

We describe the strong-forgery-attack for the naive PDVS scheme.

Select $\varepsilon \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ for accepted (m,σ), and compute $Q_A^* = Q_A - \varepsilon P_{BC}$, $Q_B^* = Q_{BC} + \varepsilon P_A$ and output forgery (Q_A^*, Q_{BC}^*, l). Then (Q_A^*, Q_{BC}^*, l) satisfies $e(Q_A^*, P_A)e(Q_{BC}^*, P_{BC}) = e(M,P)$. Therefore anyone can generate forgery (Q_A^*, Q_B^*, l) by using an acceptable message/signature pair.

4.2 Proposed PDVS Scheme

4.2.1 Idea

To prevent the strong-forgery attack in Sect 4.1.3, we add a signing procedure for generating a new part of signature ch corresponding to (m,σ). ch is computed only by using signer's or verifier's secret key. A valid signature consists of σ and ch. Even if a third party generates (m,σ^*), he cannot generate ch' corresponding to (m,σ^*). Hence a third party never generates strong-forgery (m,σ^*,ch^*).

4.2.2 PDVS Scheme

Let σ be a signature which is generated by DSign or TSim in the naive PDVS scheme and Σ be a family of σ.

SetUp: Let be the same as SetUp in the naive PDVS. Besides let $\mathcal{G}:\{0,1\}^* \times \Sigma \times \mathbb{H} \longrightarrow \mathbb{H}$ be a hash function family and G be a random member of $\mathcal{G}$.

SKeyGen : Pick $(a,a') \stackrel{R}{\leftarrow} \mathbb{Z}_q^{*2}$ and compute $P_A = aP$ and $P_{A'} = a'P$. The signer's public keys PKs are P_A and $P_{A'}$, and the secret keys SKs are a and a'.

VKeyGen: Pick $(b,b') \stackrel{R}{\leftarrow} \mathbb{Z}_q^{*2}$ and compute $P_B = bP$ and $P_{B'} = b'P$. Pick $c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and compute $P_c = cP$. The verifiers' public keys PKv are $P_B, P_{B'}$ and P_C. The secret keys SKv that the verifier keeps are b, b' and c. The secret key SKp that the proxy is given by the verifier is c.

DSign : Given m, generate σ by DSign in the naive PDVS scheme and compute $ch = G(m,\sigma,a'P_{B'})$. The original signature σ_{new} of m is (σ,ch).

TSim: Given m, generate σ' by TSim in the naive PDVS scheme and compute $ch' = G(m,\sigma',b'P_{A'})$. The dummy signature σ'_{new} of m is (σ',ch').

Decision : Let be the same as Decision in the naive PDVS scheme.

Distinction : Given an acceptable message/signature pair (m,σ,ch), if $m \neq m'$ for any m' which was issued with dummy signature before, output *valid*. Else if $(m = m') \wedge (\sigma = \sigma')$ for any message/dummy signature pair (m',σ') which was issued before, output *invalid*. Otherwise check whether $ch = G(m,\sigma,b'P_{A'})$, if it does, output *valid*.

4.3 Comparison

In this section, we compare previous DVS schemes with our proposed PDVS scheme in terms of the computational cost of the verification task for the verifier.

We describe the cost of computing modulo exponentiation as E and the cost of computing pairing calculation as P.

In previous strong DVS systems, Decision is performed only by the verifier. The cost of performing the Decision of the scheme by Saeednia *et al.* (Saeednia et al., 2004) is *3E*, and the scheme by Laguillaumie *et al.* (Laguillaumie and Vergnaud, 2004) is *E+4P*.

In our proposed PDVS scheme, the verification cost of the verifier is at most E. But this calculation is performed when only the message/signature pair (m,σ) satisfies $(m = m') \wedge (\sigma \neq \sigma')$ for any (m',σ') which the verifier issued before. In the PDVS system, indeed, the verifier need not issue any dummy signature. In this case, the verifier verifies that (m,σ) is valid immediately when he is reported that (m,σ) is acceptable by the proxy. Hence, in practice, the verification cost of the verifier is very smaller than that of previous DVS systems.

4.4 Security Proofs

4.4.1 Strong Unforgeability

We will prove that PDVS is satisfies sEUF-CMA.

Theorem 1 (Strong Unforgeability). *For any sEUF-CMA-adversary $\mathcal{A}$ in the random oracle model, with security parameter k, which has the success probability $\varepsilon = Succ^{seuf-cma}_{PDVS,\mathcal{A}}(k)$, and makes q_G queries to the random oracle, q_{Σ_S} queries to the original signing oracle, q_{Σ_T} queries to the dummy signing oracle, q_Υ queries to the Distinction oracle, there exists an adversary $\mathcal{A}$ for CDH which has the advantage $\mathsf{Succ}^{cdh}_{Gen,\mathcal{A}}(k)$ upper-bounded by ε' such that*

$$\varepsilon' \geq \varepsilon - \frac{(q_G+q_\Upsilon)(q_{\Sigma_S}+q_{\Sigma_T})}{2^{4k}} - \frac{1}{2^k}.$$

Proof. Suppose $\mathcal{A}$ is an adversary that (k,t,ε)-breaks sEUF-CMA of the PDVS scheme. $\mathcal{A}$ who is given information $params, PKs, PKv$ and SKp can query messages for original singing and dummy signing oracle and obtains signatures (σ, ch) for any message m. (m_i, σ_i, ch_i) for $i \in \{1, \cdots, q_{\Sigma_S} + q_{\Sigma_T}\}$ are message/signature pairs which $\mathcal{A}$ obtains by signing oracles. $\mathcal{A}$ also can ask the Distinction oracle whether message and the signature pairs are valid or not. Finally $\mathcal{A}$ outputs a forgery (m^*, σ^*, ch^*).

We construct $\mathcal{B}$ which solves CDH problem by using $\mathcal{A}$. Let (X,Y) be an inputs for $\mathcal{B}$ where $X = xP$ and $Y = yP$ in $\mathbb{G}$ for uniformly random (x,y) in $\mathbb{Z}_q^*$. $\mathcal{B}$ computes xyP. Let σ be a triple(Q_A, Q_{BC}, l) and Σ be a family of σ.

Input. $\mathcal{B}$ picks $(a,b,c) \overset{R}{\leftarrow} \mathbb{Z}_q^{*3}$, sets $P_A = aP, P_B = bP, P_C = cP, P_{A'} = X, P_{B'} = Y$, and inputs $P_A, P_B, P_C, P_{A'}, P_{B'}, c$ to $\mathcal{A}$.

G-Queries. For any query $(m, \sigma, \omega) \in \{0,1\}^* \times \Sigma \times \mathbb{H}$, $\mathcal{B}$ checks whether $e(\omega, P) = e(P_{A'}, P_{B'})$, if it does, $\mathcal{B}$ outputs ω and halt. Else if there exist $(m, \sigma, \omega, ch, 0, \perp)$ in G-List, $\mathcal{B}$ return ch. Otherwise $\mathcal{B}$ picks $ch \overset{R}{\leftarrow} \mathbb{H}$, returns to $\mathcal{A}$ and adds $(m, \sigma, \omega, ch, 0, \perp)$ to G-List.

DSign-Queries. For any m, $\mathcal{B}$ computes $\sigma \leftarrow \mathsf{DSign}(m)$ by using a and picks $ch \overset{R}{\leftarrow} \mathbb{H}$. If there exists $(m, \sigma, *, ch, 0, \perp)$ in G-List, $\mathcal{B}$ abort the simulation. Otherwise $\mathcal{B}$ returns (σ, ch) to $\mathcal{A}$ and add $(m, \sigma, \perp, ch, 1, DSign)$ to G-List.

TSim-Queries. For any m, $\mathcal{B}$ computes $\sigma \leftarrow \mathsf{TSim}(m)$ by using b and c, and picks $ch \overset{R}{\leftarrow} \mathbb{H}$. If there exists $(m, \sigma, *, ch, 0, \perp)$ in G-List, $\mathcal{B}$ abort the simulation. $\mathcal{B}$ picks $ch \overset{R}{\leftarrow} \mathbb{H}$ and returns (σ, ch) to $\mathcal{A}$ and adds $(m, \sigma, \perp, ch, 1, TSim)$ to G-List.

Distinction-Queries. For any (m, σ, ch), if an output of $\mathsf{Decision}(m, \sigma)$ is $reject$, $\mathcal{B}$ returns $invalid$. If there does not exist $(m, \sigma, *, ch, *, *)$, $\mathcal{B}$ returns $invalid$ and adds $(m, \sigma, \perp, ch, 0, \perp)$ Else if there exists $(m, \sigma, *, ch, 1, TSim)$ in G-List, $\mathcal{B}$ returns $invalid$. Otherwise $\mathcal{B}$ returns $valid$.

The above simulation is perfectly indistinguishable from the real forgery unless the following events happen:

- The simulation is aborted in **DSign-Queries** or **TSim-Queries**. This happens with the probability at most $(q_G + q_\Upsilon)(q_{\Sigma_S} + q_{\Sigma_T})2^{-4k}$ through the entire simulation.

If the adversary outputs strong forgery (m^*, σ^*, ch^*), $\mathcal{B}$ does not query to the random oracle, then fails to solve CDH problem. This happens with the probability at most 2^{-k}.

Thus, we obtains the following probability:

$$\varepsilon' \geq \varepsilon - \frac{(q_G + q_\Upsilon)(q_{\Sigma_S} + q_{\Sigma_T})}{2^{4k}} - \frac{1}{2^k}.$$

□

4.4.2 Privacy of Signer's Identity

We will prove that PDVS scheme satisfies PSI in the random oracle model, assuming that GBDH is hard.

Theorem 2 (Privacy of Signer's Identity). *For any PSI-CMA-adversary $\mathcal{A}$, in the random oracle model, with security parameter k, which has the success probability $\varepsilon = Succ^{psi-cma}_{PDVS,\mathcal{A}}(k)$, and makes q_H and q_G queries to the random oracle, q_{Σ_S} queries to the original signing oracle, q_{Σ_T} queries to the dummy signing oracle, q_Γ queries to the Decision oracle, q_Υ queries to the Distinction oracle, there exist an adversary $\mathcal{A}$ for GBDH which has the advantage $\mathsf{Succ}^{gbdh}_{Gen,\mathcal{A}}(k)$ upper-bounded by ε' such that*

$$\varepsilon' \geq \frac{\varepsilon}{2} - \frac{q_\Gamma + q_\Upsilon}{2^k} - \frac{(q_H + q_{\Sigma_S} + q_{\Sigma_T})(q_{\Sigma_S} + q_{\Sigma_T})}{2^k} - \frac{(q_G + q_\Upsilon)(q_{\Sigma_S} + q_{\Sigma_T})}{2^{4k}}$$

Proof. We construct $\mathcal{B}$ which solves GBDH by using $\mathcal{A}$. Let (X,Y,Z) be an inputs for $\mathcal{B}$ where $X = xP$, $Y = yP$ and $Z = zP$ in $\mathbb{G}$ for uniformly random (x,y,z) in $\mathbb{Z}_q$. $\mathcal{B}$ computes $e(P,P)^{xyz}$ by using DBDH oracle.

In order to simulate the environment of $\mathcal{A}$, $\mathcal{B}$ performs as follows:

Input. $\mathcal{B}$ picks $\alpha \overset{R}{\leftarrow} \mathbb{Z}_q^*$, $(a'_0, a'_1, b') \overset{R}{\leftarrow} \mathbb{Z}_q^{*3}$, sets $P_{A_0} = X$, $P_{A'_0} = a'_0 P$, $P_{A_1} = \alpha X$, $P_{A'_1} = a'_1 P$, $P_B = Y$, $P_{B'} = b'P$, $P_C = Z$, and inputs $P_{A_0}, P_{A'_0}, P_{A_1}, P_{A'_1}, P_B, P_{B'}$ and P_C to $\mathcal{A}$.

H-Queries. For any query $(m, v) \in \{0,1\}^* \times \mathbb{H}$

- $\mathcal{B}$ checks whether H-List includes a quadruple $(m, v, \perp, M)$. If it does, $\mathcal{B}$ returns M.
- Else $\mathcal{B}$ browses H-List and checks for all quadruple $(m, \perp, l, M)$ whether $v^{1/l} = e(P,P)^{xyz}$ by using DBDH oracle. If it does, $\mathcal{B}$ returns M.
- Otherwise, $\mathcal{B}$ picks $M \overset{R}{\leftarrow} \mathbb{Z}_q^*$, records $(m, v, \perp, M)$ in H-List, and returns M.

G-Queries. For any query $(m, Q_A, Q_{BC}, l, \omega) \in \{0,1\}^* \times \mathbb{H}^2 \times \mathbb{Z}_q^* \times \mathbb{H}$, $\mathcal{B}$ checks whether $\omega = a'_i b' P$. If it does, there exists $(m, Q_A, Q_{BC}, l, \omega, ch, 0, \perp)$ in G-List, $\mathcal{B}$ returns ch. Otherwise $\mathcal{B}$ picks $ch \overset{R}{\leftarrow} \mathbb{H}$, returns to $\mathcal{A}$ and adds $(m, Q_A, Q_{BC}, l, \omega, ch, 0, \perp)$ to G-List.

DSign-Queries (resp. TSim-Queries). For any m, whose signature is queried to Σ_{Si}(resp. Σ_{Ti}) corresponding to Signer S_i, by either the adversary or the challenger, $\mathcal{B}$ picks $(q_A, q_B) \overset{R}{\leftarrow} \mathbb{Z}_q^{*2}$, $l \overset{R}{\leftarrow} \mathbb{Z}_q^*$, and computes $M = q_A \alpha^i P_{A_i} + q_B P_B$, and sets $Q_A = q_A \alpha^i P$ and $Q_{BC} = q_B P$.

- If H-List includes a quadruple $(m, \perp, l^{\alpha^i}, *)$, $\mathcal{B}$ aborts the simulation,
- Else $\mathcal{B}$ browses H-List and checks for each quadruple $(m, v, \perp, M)$, whether $v^{1/l} = e(P,P)^{xyz}$ by using DBDH oracle. If it does, $\mathcal{B}$ aborts the simulation.
- Otherwise $\mathcal{B}$ adds the quadruple $(m, \perp, l^{\alpha^i}, M)$ to H-List and returns (Q_A, Q_{BC}, l).

$\mathcal{B}$ picks $ch \overset{R}{\leftarrow} \mathbb{H}$. If there exist $(m, Q_A, Q_{BC}, l, *, ch, 0, \perp)$ in G-List, abort the simulation. Otherwise return (Q_A, Q_{BC}, l, ch) to $\mathcal{A}$ and add $(m, Q_A, Q_{BC}, l, \perp, ch, 1, DSign_i)$ (resp.$(m, Q_A, Q_{BC}, l, \perp, ch, 1, TSim_i)$) in G-List.

DVerify-Queries. For any inputs $(m, Q_A, Q_{BC}, l, ch, S_i)$, the followings are queried

- $\mathcal{B}$ checks whether H-List includes a quadruple $(m, *, *, M)$. If it does not, $\mathcal{B}$ returns *reject*.
- If H-List includes a quadruple $(m, \perp, l, M)$, $\mathcal{B}$ returns *accept* if $e(Q_{A_i}, P_{A_i})e(Q_{BC}, P_B) = e(M, P)$.
- If H-List includes a quadruple $(m, v, \perp, M)$, $\mathcal{B}$ returns *accept* if both $v^{1/l\alpha^i} = e(P,P)^{xyz}$ and $e(Q_{A_i}, P_{A_i})e(Q_{BC}, P_B) = e(M, P)$.

Distinction-Queries. For any $(m, Q_A, Q_{BC}, l, ch, S_i)$, $\mathcal{B}$ checks whether (m, Q_A, Q_{BC}, l, ch) is acceptable or not by performing the **DVerify-Queries**, if it does not, returns *invalid*. If there does not exist $(m, Q_A, Q_{BC}, l, *, ch, *, *)$, $\mathcal{B}$ returns *invalid* and adds $(m, Q_A, Q_{BC}, l, \perp, ch, 0, \perp)$ Else if there exist $(m, Q_A, Q_{BC}, l, *, ch, 1, TSim_i)$ in G-List, return *invalid*. Otherwise $\mathcal{B}$ returns *valid*.

For m^* that $\mathcal{A}$ outputs, $\mathcal{B}$ picks $i \overset{R}{\leftarrow} \{0,1\}$ and generates $\sigma^* = (Q^*_{A_i}, Q^*_{BC}, l^*, ch^*)$ by using the above **DSign-Queries** or **TSim-Queries** of S_i. $\mathcal{B}$ returns σ^* to $\mathcal{A}$.

After receiving σ^*, $\mathcal{A}$ outputs i' . $\mathcal{B}$ obtains $(m^*, v^*, \perp, M^*)$ in H-List and outputs $C = v^{*1/l\alpha^i}$. Otherwise, $\mathcal{B}$ outputs a random element of $\mathbb{G}$.

The above simulation is perfectly indistinguishable from the real attack unless the following events happen:

- The simulation aborts in **DSign-Queries** or **TSim-Queries**. This happens with the probability at most $(q_H + q_{\Sigma_S} + q_{\Sigma_T})(q_{\Sigma_S} + q_{\Sigma_T})2^{-k} + (q_G + q_\Upsilon)(q_{\Sigma_S} + q_{\Sigma_T})2^{-4k}$ through the entire simulation.
- The valid signature of m, (Q_A, Q_{BC}, l), was generated without querying (m, u^l) to H oracle, and was queried to Γ or Υ oracle. Since $H(m, u^l)$ is uniformly distributed, this happens with the probability at most $(q_\Gamma + q_\Upsilon)2^{-k}$ through the entire simulation.

The signature σ^* provides $\mathcal{A}$ no information about i if $(m^*, v^*, \perp, M^*)$ was not queried to H-Queries. Therefore, in this case $\mathcal{A}$ succeeds with the probability 1/2 [4].

Thus,we obtains the following probability:

$$\varepsilon' \geq \frac{\varepsilon}{2} - \frac{q_\Gamma + q_\Upsilon}{2^k} - \frac{(q_H + q_{\Sigma_S} + q_{\Sigma_T})(q_{\Sigma_S} + q_{\Sigma_T})}{2^k} - \frac{(q_G + q_\Upsilon)(q_{\Sigma_S} + q_{\Sigma_T})}{2^{4k}}$$

□

4.4.3 Source Hiding

We will show that PDVS satisfies SH.

Theorem 3 (Source Hiding). *In the PDVS scheme we propose, the following expression holds.*

$$Adv^{sh}_{PDVS,\mathcal{A}}(k) = 0$$

Proof. We prove the following fact. Given public keys of $P_A, P_{A'}, P_B, P_{B'}$ and P_C, secret keys of a, a', b, b' and c, arbitrary message m^*, and signature for m^*, $(Q^*_A, Q^*_{BC}, l^*, ch^*)$, $\mathcal{A}$ can not distinguish by which procedure of DSign or TSim $(Q^*_A, Q^*_{BC}, l^*, ch^*)$ is generated.

For $N \in \mathbb{G}$ in DSign and $N' \in \mathbb{G}$ in TSim, there exists $n, n' \in Z^*_q$ such that $N = nP, N' = n'P$.

Using this arbitrary n and n', we prove that Q_A, Q_{BC}, Q'_A and Q'_{BC} have the same distribution. Since r in DSign and r' in TSim are random values in $\{1, ..., q-1\}$, $Q_{BC} = rP$ and $Q'_A = r'P$ have the uniform distribution on the set $\{P, ..., (q-1)P\}$.

Let $f(r) := a^{-1}\{n - r(b+c)\}$, then $Q_A = a^{-1}(N - rP_{BC})$ describes $Q_A = f(r) \cdot P$. Since $f(r)$ is bijective, $f(r)$ has the uniform distribution on the set $\{1, ..., q-1\}$. So Q_A has the uniform distribution on the set $\{P, ..., (q-1)P\}$. Similarly, let $f'(r') := (b + c)^{-1}(n' - r \cdot a)$, then $Q'_{BC} = a^{-1}(N - r'P_A)$ describes $Q'_{BC} = f'(r') \cdot P$. Since $f'(r')$ is bijective, $f'(r')$ has the uniform distribution on the set $\{1, ..., q-1\}$. So Q'_{BC} has the uniform distribution on the set $\{P, ..., (q-1)P\}$. Therefore Q_A, Q_{BC}, Q'_A and Q'_{BC} have the same distribution. Moreover values of Q_A, Q_{BC}, Q'_A and Q'_{BC} depend on a random values r or r'. Hence, it is not distinguished whether a triple Q^*_A, Q^*_{BC}, l^* is generated by DSign or TSim. Besides, $ch^* = G(m^*, Q^*_A, Q^*_{BC}, l^*, a'P_{B'}) = G(m^*, Q^*_A, Q^*_{BC}, l^*, b'P_{A'})$, so it is also not distinguished whether ch^* is generated by DSign or TSim.

Therefore even if the values of all secret keys a, a', b, b' and c are revealed, it is not distinguished whether a signature is generated by DSign or TSim procedures. □

[4] ch^* is given by random oracle and does not depend on any secret keys. So ch^* does not give any information of S_i to $\mathcal{A}$.

4.4.4 Non-coincidental Property

We will show that PDVS satisfies NCP.

We consider the probability that $\sigma = \sigma'$ where $\sigma \leftarrow \mathsf{DSign}(m, SKs, PKv), \sigma' \leftarrow \mathsf{TSim}(m, SKv, SKp, PKs)$ in the random oracle model. We represent an original signature as $\sigma = (Q_A, Q_{BC}, l)$ and a dummy signature as $\sigma' = (Q'_A, Q'_{BC}, l')$. We also denote that $r \in \mathbb{Z}_q^*$ is a random string the signer selects and $r' \in \mathbb{Z}_q^*$ is a random string the verifier selects. $\mathsf{Pr}[\sigma = \sigma'] = \mathsf{Pr}[l = l'] \cdot \mathsf{Pr}[Q_A, Q_{BC} = Q'_A, Q'_{BC} | l = l'] = (q-1)^{-2}$. Hence, $\mathsf{Pr}[\sigma = \sigma']$ is negligible.

5 CONCLUSIONS

In this paper, we proposed concepts and definitions of the PDVS that a verifier can delegate some computational cost of the verification to the proxy. We defined new security requirements for the PDVS, and proposed a concrete PDVS scheme. Finally we proved that our PDVS scheme satisfies all security requirements for the PDVS under CDH and GBDH assumptions.

REFERENCES

An, J., Dodis, Y., and Rabin, T. (2002). On the security of joint signature and encryption. In *Advances in Cryptology — EUROCRYPT 2002*. Springer.

Baek, J., Safavi-Naini, R., and Susilo, W. (2005). Universal designated verifier signature proof (or how to efficiently prove knowledge of a signature). In *Advances in Cryptology — ASIACRYPT 2005*. Springer.

Chaum, D. and van Antwerpen, H. (1990). Undeniable signatures. In *Advances in Cryptology — CRYPTO 1989*. Springer.

Jakobsson, M., Sako, K., and Impagliazzo, R. (1996). Designated verifier proofs and their applications. In *Advances in Cryptology — EUROCRYPT 1996*. Springer.

Laguillaumie, F. and Vergnaud, D. (2004). Multi-designated verifiers signatures. In *International Conference on Information and Communications Security — ICICS 2004*. Springer.

Laguillaumie, F. and Vergnaud, D. (2005). Designated verifier signatures: Anonymity and efficient construction from *any* bilinear map. In *Security in Communication Networks — SCN 2004*. Springer.

Lipmaa, H., Wang, G., and Bao, F. (2005). Designated verifier signature schemes: Attacks, new security notions and a new construction. In *Automata, Languages and Programming — ICALP 2005*. Springer.

Rivest, R., Shamir, A., and Tauman, Y. (2001). How to leak a secret. In *Advances in Cryptology — ASIACRYPT 2001*. Springer.

Saeednia, S., Kremer, S., and Markowitch, O. (2004). An efficient strong designated verifier signature scheme. In *Information Security and Cryptology—ICISC 2003*. Springer.

Shahandashti, S. and Safavi-Naini, R. (2008). Construction of universal designated-verifier signatures and identity-based signatures from standard signatures. In *Public Key Cryptography — PKC 2008*. Springer.

Steinfeld, R., Bull, L., Wang, H., and Pieprzyk, J. (2003). Universal designated-verifier signatures. In *Advances in Cryptology — ASIACRYPT 2003*. Springer.

Steinfeld, R., Wang, H., and Pieprzyk, J. (2004). Efficient extension of standard schnorr/rsa signatures into universal designated-verifier signatures. In *Public Key Cryptography — PKC 2004*. Springer.

Wang, G. (2005). Designated-verifier proxy signature schemes. In *Security and Privacy in the Age of Ubiquitous Computing (IFIP/SEC 2005)*. Springer.