On Designated Verifier Signature Schemes

Michal Rjaško and Martin Stanek

Department of Computer Science
Faculty of Mathematics, Physics and Informatics
Comenius University
Mlynská dolina, 842 48 Bratislava, Slovak Republic
{rjasko,stanek}@dcs.fmph.uniba.sk

Abstract. Designated verifier signature schemes allow a signer to convince only the designated verifier that a signed message is authentic. We define attack models on the unforgeability property of such schemes and analyze relationships among the models. We show that the no-message model, where an adversary is given only public keys, is equivalent to the model, where an adversary has also oracle access to the verification algorithm. We also show a separation between no-message model and chosen-message model, where an adversary has access to the signing algorithm. Furthermore, we present a modification of the Yang-Liao designated verifier signature scheme and prove its security. The security of the modified scheme is based on the computational Diffie-Hellman problem, while the original scheme requires strong Diffie-Hellman assumption.

1 Introduction

Standard digital signature schemes allow a signer to sign messages, so that everyone can verify the authenticity of the messages. In addition, digital signature schemes have a non-repudiation property, which guarantees that a signer cannot lie about messages he has signed. However, sometimes the non-repudiation property is not needed in practice. Moreover there are cases where this property is undesired. For example the signer should not present signatures to other parties when signing personal health records, income summary, or an offer of a tenderer in e-auction system. Jakobsson et al. [4] introduced the concept of designated verifier signature (DVS for short) schemes which make it possible to a signer A to convince the designated verifier B that a message he signed is authentic. However, neither A nor the designated verifier B can convince any other party of authenticity of that message. This is achieved by ability of B to generate indistinguishable signatures from those produced by A. Thus no one else can tell whether the signature was created by A or B.

Even though the signer ambiguity exists in the designated verifier signature schemes, it does not prevent an attacker E eavesdropping on the line between A and B to get the signature before B receives it. In this case, E can be convinced with high probability that the signature was issued by A. To overcome this problem, Jakobsson et al. [4] defined a notion of strong designated verifier. In strong designated verifier signature scheme everyone can generate signatures indistinguishable (for everyone except B) from those produced by A. The strong DVS property can be achieved by encrypting the designated verifier signature with B's public key. This makes the signature indistinguishable from a random string.

Our contribution. Besides the (strong) designated verifier property, DVS schemes should provide unforgeability, a property known from the standard digital signature schemes. This property guarantees, that nobody except A and B can forge a valid signature of some (previously unsigned) message. We analyze several different models of unforgeability for the DVS schemes. Based on the resources accessible to an attacker, we differentiate between these models:

- No-message attack – an adversary is given only public keys of A and B.
- Chosen-message attack – an adversary is given public keys and access to the signing oracle.
- Check attack – an adversary is given public keys and access to the B's verification oracle.

The first two models are well known for the standard digital signatures. The check attack model can be defined only in the DVS settings.

In Section 3 we analyze relationships among these models. In particular, we show that the check model is equivalent to the no-message model. We also present a separation between the no-message model and chosen-message model. We show that this separation holds in both standard and strong DVS setting.

In Section 4 we analyze the Yang-Liao scheme [13], which is provably unforgeable in the no-message model under the strong Diffie-Hellman (DH for short) assumption. We modify this scheme so that the modified scheme is provably unforgeable in the check model under the computational DH assumption. Hence, we weaken the assumption needed for provable security of the scheme. Moreover, we show that the modified Yang-Liao scheme has the strong designated verifier property. The modified scheme and its analysis use the results of Cash et al. [1] regarding the twin DH problem (the equivalence of the strong variant of twin DH problem and the computational DH problem).

Related work. To date, numerous designated verifier signature schemes have been proposed, such as [4, 6, 10, 13]. There are also several modifications of the standard designated verifier signature schemes like universal DVS schemes [3, 11, 12, 14] and multi-designated verifier signature schemes [2, 5, 9].

Formal definitions of security notions for the DVS schemes are considered in [7, 8]. They formally define unforgeability, non-delegatability, non-transferability and disavowability. The non-delegatability notion guarantees that A and B cannot delegate their ability to generate signatures to the third party without giving the third party their secret keys. The non-transferability is another name for the designated verifier property defined in [4, 10] or also in this paper. The disavowability guarantees, that A cannot disavow signatures issued by itself (with respect to the designated verifier B).

Cash et al. [1] define the twin DH problem and its variants. They show that the strong twin DH problem is equivalent to the computational DH problem. Thus, one can modify various schemes based on the strong DH assumption, so that the resulting scheme is secure under the computational DH assumption. As an example they modify the ElGamal encryption scheme, so that the resulting scheme is secure under the computational DH assumption in the random oracle model.

2 Preliminaries

Let H be a hash function with range $\mathbb{Z}_q$. By $x \overset{\$}{\leftarrow} S$ we denote a uniform random selection of x from the set S. Concatenation of two binary strings $m_1, m_2 \in \{0,1\}^*$ is denoted as $m_1 \,||\, m_2$. We assume that from $m_1||m_2$ it is possible to uniquely determine the strings m_1 and m_2 for all $m_1, m_2 \in \{0,1\}^*$.

A function f is negligible if for every polynomial $p(\cdot)$ there exists N such that for every $n > N$ it holds that $f(n) < \frac{1}{p(n)}$. Negligible functions are denoted as $\mathrm{negl}(\cdot)$.

The Diffie-Hellman assumption. Let G be a group of order q, g be a generator of G and $x, y \in \mathbb{Z}_q$. Let $\mathrm{dh}(X, Y)$ denote a solution for Diffie-Hellman problem, i.e.

$$\mathrm{dh}(X, Y) := Z, \qquad \text{where } X = g^x, Y = g^y \text{ and } Z = g^{xy}.$$

Let $\mathrm{dhp}(X, Y, Z)$ denote a predicate $\mathrm{dhp}(X, Y, Z) := (\mathrm{dh}(X, Y) \overset{?}{=} Z)$. Standard Diffie-Hellman assumptions are assumptions about hardness of the following problems:

- Decisional DH – distinguishing between tuples (g^x, g^y, g^{xy}) and (g^x, g^y, g^z) for $z \overset{\$}{\leftarrow} \mathbb{Z}_q$
- Computational DH – computing $\mathrm{dh}(X, Y)$ for random X, Y
- Strong DH – computing $\mathrm{dh}(X, Y)$ for random X, Y, with oracle access to the predicate $\mathrm{dhp}(X, \cdot, \cdot)$
- Gap DH – computing $\mathrm{dh}(X, Y)$ for random X, Y, with oracle access to the predicate $\mathrm{dhp}(\cdot, \cdot, \cdot)$

By hardness we mean impossibility to solve the problems with non-negligible probability in the probabilistic polynomial time. Note the difference between strong DH problem and gap DH, where an adversary is given access to the full decision oracle $\mathrm{dhp}(\cdot, \cdot, \cdot)$ (instead of decision oracle with the first input fixed – $\mathrm{dhp}(X, \cdot, \cdot)$). Clearly, the computational DH is the weakest assumption and the gap DH is the strongest assumption. A natural ambition is to construct cryptographic schemes that depend on the weakest possible assumptions.

The Twin Diffie-Hellman assumption. Recently, Cash, Kiltz and Shoup [1] defined a so called Twin Diffie-Hellman problem, which is equivalent in its "strong" variant to the computational DH problem. Using this fact, one can modify various schemes (such as hashed ElGamal), so that they can be proved to be secure under computational DH assumption, instead of strong DH assumption.

Let $2\mathrm{dh}(X_1, X_2, Y)$ denote a solution for twin Diffie-Hellman problem, i.e.

$$2\mathrm{dh}(X_1, X_2, Y) := (\mathrm{dh}(X_1, Y), \mathrm{dh}(X_2, Y)).$$

By $2\mathrm{dhp}(X_1, X_2, Y, Z_1, Z_2)$ we denote a predicate

$$2\mathrm{dhp}(X_1, X_2, Y, Z_1, Z_2) := (2\mathrm{dh}(X_1, X_2, Y) \overset{?}{=} (Z_1, Z_2)).$$

We consider several twin Diffie-Hellman assumptions about hardness of the following problems:

- Decisional twin DH – distinguishing between the following tuples: $(g^{x_1}, g^{x_2}, g^y, g^{x_1 y}, g^{x_2 y})$ and $(g^{x_1}, g^{x_2}, g^y, g^{z_1}, g^{z_2})$ for $z_1, z_2 \overset{\$}{\leftarrow} \mathbb{Z}_q$
- Twin DH – computing $2\mathrm{dh}(X_1, X_2, Y)$ for random X_1, X_2, Y
- Strong twin DH – computing $2\mathrm{dh}(X_1, X_2, Y)$ for random X_1, X_2, Y, with oracle access to the predicate $2\mathrm{dhp}(X_1, X_2, \cdot, \cdot, \cdot)$

The main result of [1] is stated in the following proposition.

Proposition 1. *The computational DH assumption holds if and only if the strong twin DH assumption holds.*

3 Security definitions

A designated verifier signature scheme is a protocol with two participants Alice (A) and Bob (B). Alice wants to construct a signature σ of some message m, such that only Bob can verify the validity of σ. Moreover, Bob cannot convince any other party of σ's validity. The designated verifier property is thus an ability of Bob to generate indistinguishable signatures of m, so that no one else can distinguish, whether the signature σ is created by A or B.

Definition 1 (Designated verifier signature scheme). *Let M be some fixed message space (usually $M = \{0,1\}^*$). A designated verifier signature scheme (DVSS) for two participants A and B is a tuple $\langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}\rangle$ of polynomial algorithms, where*

- *$\mathit{Gen}(1^n)$ is a probabilistic algorithm, which takes on input a security parameter 1^n and outputs a tuple of keys (pk_A, sk_A, pk_B, sk_B), where pk_A, sk_A are public and private keys of A and pk_B, sk_B are public and private keys of B.*
- *$\mathit{Sig}(m, sk_A, pk_B)$ is a probabilistic algorithm that takes on input a message $m \in M$, A's private key and B's public key, and outputs a signature σ, denoted as $\sigma \leftarrow \mathit{Sig}(m, sk_A, pk_B)$.*
- *$\mathit{Vrf}(m, \sigma, sk_B, pk_A)$ is a deterministic algorithm, which takes on input a message m, signature σ, B's private key and A's public key. It outputs a bit $b = 1$ if σ is a valid signature of m, otherwise it outputs $b = 0$. If the message m is not needed for the verification procedure we say that the scheme has "message recovery" property (we'll see example of such a scheme in the Section 4).*
- *$\mathit{Sim}(m, pk_A, sk_B)$ is a probabilistic algorithm producing signatures indistinguishable from those produced by $\mathit{Sig}(\cdot, sk_A, pk_B)$. In some cases (e.g. the definition of strong designated verifier) we need the Sim algorithm with different set of inputs – instead of the secret key sk_B, it takes B's public key pk_B. We denote this modified algorithm as $\mathit{Sim}'(m, pk_A, pk_B)$.*

In addition, for every n, every tuple (pk_A, sk_A, pk_B, sk_B) generated by $\mathit{Gen}(1^n)$, and every message $m \in M$, it must hold that

$$\mathit{Vrf}(m, \mathit{Sig}(m, sk_A, pk_B), sk_B, pk_A) = 1.$$

The public and secret keys of A and B can be (and usually are) generated independently. That is, the algorithm *Gen* can be "divided" into two independent algorithms $\mathit{Gen}_A(1^n) \rightarrow (pk_A, sk_A)$ and $\mathit{Gen}_B(1^n) \rightarrow (pk_B, sk_B)$. Hence, A and B need not to communicate in order to obtain public and secret keys. To simplify the presentation, we use the notation with only one algorithm *Gen*.

Saednia, Kremer, Markowitch [10] define the designated verifier property as follows.

The designated verifier experiment $\mathit{Ind}\text{-}\mathit{DV}_{D,\Pi}(n)$:

1. *Run $\mathit{Gen}(1^n)$ to obtain (pk_A, sk_A, pk_B, sk_B).*
2. *A random bit $b \overset{\$}{\leftarrow} \{0,1\}$ is chosen.*
3. *Adversary D is given pk_A, pk_B and oracle access to either $\mathit{Sig}(\cdot, sk_A, pk_B)$ if $b = 0$ or to $\mathit{Sim}(\cdot, pk_A, sk_B)$ if $b = 1$. Eventually, D outputs a bit b'.*
4. *The output of the experiment is 1 if $b = b'$ and 0 otherwise.*

Definition 2 (Designated verifier). *Let $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}\rangle$ be a DVSS for A and B. We say that Π has a designated verifier property if for any probabilistic polynomial time adversary D there exists a negligible function negl, such that*

$$\Pr[\mathit{Ind}\text{-}\mathit{DV}_{D,\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

We note that there exist schemes for which $\mathit{Sim}(\cdot, pk_A, sk_B)$ produces identically distributed signatures as those produced by $\mathit{Sig}(\cdot, sk_A, pk_B)$. That is, no distinguisher (running in an arbitrary time) can win the *Ind-DV* experiment with probability greater than $\frac{1}{2}$. We say that such schemes has *perfect* designated verifier property. When a scheme has this property, then no one can tell whether a signature was issued by A or B, even if their secret keys are compromised (given public keys, the distinguisher running in an arbitrary time can find secret keys by itself).

Sometimes, a potential attacker Eve can be convinced (with high probability), that the signature was created by Alice. For example, when Eve intercepts a signature (m, σ) sent over a network, which Bob could not see yet. In this case, we need a stronger notion of designated verifier.

The strong DV experiment $\mathit{Ind\text{-}SDV}_{D,\Pi}(n)$:

1. *Run* $\mathit{Gen}(1^n)$ *to obtain* (pk_A, sk_A, pk_B, sk_B).
2. *A random bit* $b \xleftarrow{\$} \{0,1\}$ *is chosen.*
3. *Adversary* D *is given* pk_A, pk_B *and oracle access to either* $\mathit{Sig}(\cdot, sk_A, pk_B)$ *if* $b = 0$ *or to* $\mathit{Sim}'(\cdot, pk_A, pk_B)$ *if* $b = 1$. *Eventually,* D *outputs a bit* b'.
4. *The output of the experiment is* 1 *if* $b = b'$ *and* 0 *otherwise.*

Definition 3 (Strong desig. verifier). *Let* $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}' \rangle$ *be a DVSS for* A *and* B. *We say that* Π *has a strong designated verifier property if for any probabilistic polynomial time adversary* D *there exists a negligible function negl, such that*

$$\Pr[\mathit{Ind\text{-}SDV}_{D,\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

Thus if the scheme Π has the strong designated verifier property, then everyone is able to generate indistinguishable signatures using the algorithm Sim'. The strong designated verifier property is usually achieved by encrypting the signature (with B's public-key). This makes the signature indistinguishable from a random string.

Besides the designated verifier property, the designated verifier signature scheme should also guarantee, that nobody except Alice and Bob can produce a valid signature for some message m, i.e. a potential attacker Eve is unable to construct a pair (m, σ), where σ is a valid signature of m. Based on the resources accessible to the attacker, Eve can perform an attack in several different models:

- no-message attack – Eve has access to public keys and parameters of participants Alice and Bob (this knowledge is assumed in all other models as well)
- chosen-message attack – Eve has oracle access to Alice's signing algorithm
- check attack – Eve has oracle access to Bob's verification algorithm

While the first two models are well known from the standard digital signature schemes, the check model is meaningful only in the designated verifier settings. The no-message model is the most used approach in the proofs of security of current DVSS [6, 10, 13].

The no-message experiment $\mathit{NMA\text{-}forge}_{E,\Pi}(n)$:

1. *Run* $\mathit{Gen}(1^n)$ *to obtain* (pk_A, sk_A, pk_B, sk_B).
2. *Adversary* E *is given* pk_A, pk_B *and outputs* (m, σ).
3. *The output of the experiment is a bit* $b = \mathit{Vrf}(m, \sigma, sk_B, pk_A)$.

Definition 4 (Unforgeability in the no-message model).
A DVSS $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim} \rangle$ is unforgeable in the no-message model, if for all probabilistic polynomial-time adversaries E, there exists a negligible function negl *such that:*

$$\Pr[\mathit{NMA\text{-}forge}_{E,\Pi}(n) = 1] \leq \mathrm{negl}(n).$$

In the chosen-message attack, an adversary E has oracle access to the signing algorithm. He can adaptively ask queries to its oracle and eventually outputs a pair (m, σ), such that m was not queried to the signing oracle.

The chosen-message experiment $\mathit{CMA\text{-}forge}_{E,\Pi}(n)$:

1. *Run* $\mathit{Gen}(1^n)$ *to obtain* (pk_A, sk_A, pk_B, sk_B).
2. *Adversary E is given public keys pk_A, pk_B and oracle access to $\mathit{Sig}(\cdot, sk_A, pk_B)$. The adversary then outputs (m, σ). Let Q be the set of all queries made by E to the signing oracle.*
3. *The output of the experiment is* 1, *if* $\mathrm{Vrf}(m, \sigma, sk_B, pk_A) = 1 \wedge m \notin Q$, *and* 0 *otherwise.*

Definition 5 (Unforgeability in the chosen-message model).
A DVSS $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim} \rangle$ is unforgeable in the chosen-message model, if for all probabilistic polynomial-time adversaries E, there exists a negligible function negl *such that:*

$$\Pr[\mathit{CMA\text{-}forge}_{E,\Pi}(n) = 1] \leq \mathrm{negl}(n).$$

In the following check experiment, an adversary E has oracle access to the verification algorithm. It means that E can ask whether (m, σ) is a valid "message, signature" pair.

The check experiment $\mathit{Check\text{-}forge}_{E,\Pi}(n)$:

1. *Run* $\mathit{Gen}(1^n)$ *to obtain* (pk_A, sk_A, pk_B, sk_B).
2. *Adversary E is given public keys pk_A, pk_B and oracle access to* $\mathrm{Vrf}(\cdot, \cdot, sk_B, pk_A)$. *The adversary then outputs (m, σ).*
3. *The output of the experiment is a bit* $b = \mathrm{Vrf}(m, \sigma, sk_B, pk_A)$.

Definition 6 (Unforgeability in the check model).
A DVSS $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim} \rangle$ is unforgeable in the check model, if for all probabilistic polynomial-time adversaries E, there exists a negligible function negl *such that:*

$$\Pr[\mathit{Check\text{-}forge}_{E,\Pi}(n) = 1] \leq \mathrm{negl}(n).$$

3.1 Relationships among models

Clearly, if some scheme Π is unforgeable in the check model, then it is unforgeable in the no-message model. The opposite direction is a little bit more involved.

Theorem 1. *Let $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim} \rangle$ be a DVSS. If Π is unforgeable in the no-message model, then it is unforgeable in the check model.*

Proof. Let E be a polynomial adversary and let

$$\varepsilon(n) = \Pr[\mathit{Check\text{-}forge}_{E,\Pi}(n) = 1].$$

Our goal is to prove that if Π is unforgeable in the no-message model, then $\varepsilon(n)$ is negligible. Let $q(n)$ be the maximum number of queries to the verification oracle, which E asks during its attack. Without loss of generality we can assume that E outputs only those signatures he verified by asking the verification oracle. Consider the following algorithm D performing attack in the no-message model:

Algorithm D:
The algorithm is given (pk_A, pk_B)

1. Choose $i \xleftarrow{\$} \{0, \ldots, q(n) - 1\}$. The value i represents D's guess of E's first valid query to the verification oracle. Note that if E forges a signature, then such valid query must exist.
2. Given pk_A and pk_B simulate the adversary E in the Check-forge experiment. If E asks the verification oracle j-th query (m_j, σ_j) $(j = 0, 1, \ldots, q(n) - 1)$, answer it in the following way:
   - if $j \neq i$ answer 0
   - otherwise output (m_j, σ_j) and terminate the execution.
3. If the simulation of E ends and D hasn't returned some signature yet, return $\perp$. (In this case, the guess i of the first valid query was too "big", i.e. E asked less than i queries.)

It is obvious that D runs in a polynomial time. The success probability of D is given by the probability that the first valid query of E has index i. Let $\mathit{FirstValid}(j)$ denote the event that E's j-th query is the first valid query of E. Hence,

$$\varepsilon(n) \leq \sum_{j=1}^{q(n)} \Pr\left[\mathit{FirstValid}(j)\right] \tag{1}$$

On the other hand,

$$\begin{aligned}\Pr[\mathit{NMA\text{-}forge}_{D,\Pi}(n) = 1] &= \Pr\left[i \xleftarrow{\$} \mathbb{Z}_{q(n)}; \mathit{FirstValid}(i)\right] = \\ &= \sum_{j=1}^{q(n)} \Pr\left[i \xleftarrow{\$} \mathbb{Z}_{q(n)}; \mathit{FirstValid}(i) \wedge i = j\right] = \\ &= \sum_{j=1}^{q(n)} \Pr\left[i \xleftarrow{\$} \mathbb{Z}_{q(n)}; \mathit{FirstValid}(j) \wedge i = j\right] = \\ &= \sum_{j=1}^{q(n)} \Pr\left[\mathit{FirstValid}(j)\right] \cdot \frac{1}{q(n)} \geq \frac{\varepsilon(n)}{q(n)},\end{aligned}$$

where we used the fact that

$$\mathit{FirstValid}(i) \wedge i = j \;\Leftrightarrow\; \mathit{FirstValid}(j) \wedge i = j,$$

and the last inequality is the consequence of (1). Thus,

$$\varepsilon(n) \leq q(n) \cdot \Pr[\textit{NMA-forge}_{D,\mathit{\Pi}}(n) = 1].$$

By the assumption that $\mathit{\Pi}$ is unforgeable in the no-message model we conclude that $\varepsilon(n)$ is negligible. □

Remark 1. The reduction in the proof above is not "tight", since the adversary D has much smaller probability of success than the adversary E. The concrete security of the reduction can be obtained easily. Fix some security parameter n. We say that some scheme $\mathit{\Pi}$ is (t, ε) unforgeable in the check model, if for any adversary E running in time t (relative to some fixed model of computation) it holds

$$\Pr[\textit{Check-forge}_{E,\mathit{\Pi}}(n) = 1] \leq \varepsilon.$$

Similarly we can define (t, ε) unforgeability in the no-message model. Then, in the concrete security settings, we can rewrite the Theorem 1 as follows:

Proposition 2. *Let $\mathit{\Pi} = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}\rangle$ be a DVSS. If $\mathit{\Pi}$ is (t, ε) unforgeable in the no-message model, then it is $(t + c, \varepsilon \cdot q_t)$ unforgeable in the check model, where c is a small constant and q_t is a maximum number of queries which can be asked by an adversary running in time t ($q_t \leq t$).*

The constant c is a time overhead that D needs to simulate E. It's value and the value of q_t depend on a model of computation (which we do not specify here).

In the following theorem we show a separation between the no-message model and chosen-message model.

Theorem 2. *Let $\mathit{\Pi} = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}\rangle$ be a DVSS unforgeable in the no-message model. Then there exists a DVSS $\mathit{\Pi}'$, which is unforgeable in the no-message model, but insecure in the chosen-message model.*

Proof. Let $\mathit{\Pi}' = \langle \mathit{Gen}', \mathit{Sig}', \mathit{Vrf}', \mathit{Sim}'\rangle$ be a DVSS defined as

- $\mathit{Gen}'(1^n) = \mathit{Gen}(1^n)$,
- $\mathit{Sig}'(m', sk_A, pk_B) = \mathit{Sig}(m, sk_A, pk_B)$,
- $\mathit{Vrf}'(m', \sigma, sk_B, pk_A) = \mathit{Vrf}(m, \sigma, sk_B, pk_A)$,
- $\mathit{Sim}'(m', pk_A, sk_B) = \mathit{Sim}(m, pk_A, sk_B)$,

where $m' = b_1 \cdots b_l$ and $m = 0b_2 \cdots b_l$, $b_i \in \{0, 1\}$. Hence, the algorithm Sig' produces the same signature for messages $b_1 \cdots b_l$ and $\overline{b_1} b_2 \cdots b_l$. It is easy to see that such scheme is correct

$$\mathit{Vrf}'(m', \mathit{Sig}'(m', sk_A, pk_B), sk_B, pk_A) = \mathit{Vrf}(m, \mathit{Sig}(m, sk_A, pk_B), sk_B, pk_A) = 1.$$

Assume that $\mathit{\Pi}$ is unforgeable in the no-message model. We show that also $\mathit{\Pi}'$ is unforgeable in the no-message model. Let E' be a polynomial adversary, which can find a valid message-signature pair $(m', \sigma'); m' = b_1 \cdots b_l$ for scheme $\mathit{\Pi}'$ and consider an adversary E, which simulates E' and then outputs (m, σ'), where $m = 0b_2 \cdots b_l$. By the definition of the scheme $\mathit{\Pi}'$ we know, that

$$\mathit{Vrf}'(m', \sigma', sk_B, pk_A) = \mathit{Vrf}(m, \sigma', sk_B, pk_A),$$

thus

$$\Pr[\mathit{NMA\text{-}forge}_{E,\Pi}(n) = 1] \geq \Pr[\mathit{NMA\text{-}forge}_{E',\Pi'}(n) = 1].$$

Hence, if Π is unforgeable in the no-message model, then so is Π'. However, Π' is completely insecure in the chosen-message model. An adversary D can query its signing oracle for a signature σ of some message m and output (m', σ), where m' is the same as m but with the first bit inverted. Clearly, such an adversary D succeeds in forging a signature in the chosen-message model against Π'. □

Remark 2. Note that the separation between the no-message and chosen-message models holds also for the strong DVS setting. Let $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}' \rangle$ be a strong DVSS and let $\Pi' = \langle \mathit{Gen}', \mathit{Sig}', \mathit{Vrf}', \mathit{Sim}'' \rangle$ be defined exactly as above, except the algorithm Sim'':

$$\mathit{Sim}''(m', pk_A, pk_B) = \mathit{Sim}'(m, pk_A, pk_B),$$

where $m' = b_1 \cdots b_l$ and $m = 0b_2 \cdots b_l$, $b_i \in \{0, 1\}$. Note that the algorithms Sim' and Sim'' are different from the ones in the proof above. Instead of B's secret key sk_B, Sim' and Sim'' are given on input B's public key pk_B. If there is an adversary D', which can distinguish $\mathit{Sig}'(\cdot, sk_A, pk_B)$ from $\mathit{Sim}''(\cdot, pk_A, pk_B)$, then the following adversary D distinguishes between $\mathit{Sig}(\cdot, sk_A, pk_B)$ and $\mathit{Sim}'(\cdot, pk_A, pk_B)$. The adversary D simulates D'. If D' asks a query $m' = b_1 \cdots b_l$, the adversary D asks its oracle a query $m = 0b_2 \cdots b_l$ and returns the answer to D'. Clearly, the adversary D' cannot detect any difference between the simulation and the strong DV experiment. Hence,

$$\Pr[\mathit{Ind\text{-}SDV}_{D,\Pi}(n) = 1] \geq \Pr[\mathit{Ind\text{-}SDV}_{D',\Pi'}(n) = 1].$$

This means that Π' is a strong DVSS, if Π is a strong DVSS.

4 Yang-Liao scheme

We review the Yang-Liao [13] DVSS ("YL scheme" for short) and then we present its modification, such that the resulting scheme is secure in the check model under the computational DH assumption.

YL scheme [13]
Let G be a group of prime order q, where G is a subgroup of $\mathbb{Z}_p^*$ for large primes p, q (q is a prime factor of $p - 1$). Let g be a generator of G. Define a designated verifier signature scheme as follows:

- $\mathit{Gen}(1^n)$ generates a tuple (x, X, y, Y) of private and public keys:
  - A: private key $x \overset{\$}{\leftarrow} \mathbb{Z}_q$, public key $X = g^x$
  - B: private key $y \overset{\$}{\leftarrow} \mathbb{Z}_q$, public key $Y = g^y$
- $\mathit{Sig}(m, x, Y)$ outputs $\langle r, s \rangle$, where
  - $s = H(m \,||\, t)$, where $t \overset{\$}{\leftarrow} \mathbb{Z}_q^*$
  - $r = (m \,||\, t) \cdot Y^{xs}$
- $\mathit{Vrf}(m, \langle r, s \rangle, y, X)$:
  - $m \,||\, t \leftarrow r \cdot X^{-ys}$
  - return $(H(m \,||\, t) \overset{?}{=} s)$

- *Sim*(m, X, y) outputs $\langle r, s\rangle$, where
  - $s = H(m \,||\, t)$, where $t \xleftarrow{\$} \mathbb{Z}_q^*$
  - $r = (m \,||\, t) \cdot X^{ys}$

It is easy to see that the scheme is correct. An obvious drawback of the scheme is, that it allows to sign messages only of length $|m| \leq \lg(p/q)$. On the other hand the scheme has the "message recovery" property, what means that the message m is not needed for the verification procedure. Security of the scheme in the no-message model is given in the following proposition.

Proposition 3 (Yang-Liao [13]). *If an adversary is able to forge a valid designated verifier signature with non-negligible probability and H is modeled as a random oracle, then strong DH problem can be solved with non-negligible probability.*

For discussion on the designated verifier property and proof of the above proposition, see [13]. Note that the security of the scheme in the no-message model is based on the strong DH assumption.

From the Theorem 1 and the proposition above it follows that if H is a random oracle, then an ability to forge signature in the check model leads to an ability of solving the strong DH problem. However, the reduction in the proof of the Theorem 1 is not tight. In the following proof we present a tight reduction.

Theorem 3. *If the strong DH problem is hard relative to G, and H is modeled as a random oracle, then the YL scheme is unforgeable in the check model.*

Proof. (sketch) Let $\Pi = \langle Gen, Sig, Vrf, Sim\rangle$ be the YL scheme. Let E be a polynomial adversary attacking Π in the check model. We construct an algorithm D, which has oracle access to $\mathrm{dhp}(X, \cdot, \cdot)$ and solves a strong DH problem whenever E forges a valid signature:

Algorithm D:

The algorithm is given (G, p, q, g, X, Y), where $X = g^x$, $Y = g^y$ and has oracle access to $\mathrm{dhp}(X, \cdot, \cdot)$.

1. Given E public keys X and Y. Maintain a hash table of all E's queries to the random oracle H. The table consists of tuples (m, t, s) and is initially empty.
2. When E asks its random oracle query $H(m||t)$, answer in the following way
   - search the table for entry (m, t, s), if such entry exists, output s,
   - otherwise choose $s \xleftarrow{\$} \mathbb{Z}_q$, store (m, t, s) in the table and output s.
3. When E makes a query to its verification oracle *Vrf*(r, s), answer as follows:
   - search the table for entry (m, t, s), if no such entry exists return 0,
   - otherwise compute $Z = (r/(m||t))^{1/s} \mod p$ and check, whether $\mathrm{dhp}(X, Y, Z)$ holds:
     - If so, output Z as the solution of the strong DH and terminate the execution, otherwise return 0 as an answer to E's verification oracle query.
4. At the end of E's execution, it outputs a pair (r, s). Check if it is a valid signature similarly as in the previous step. If (r, s) is a valid signature, output Z, otherwise output $\perp$.

Clearly, E's view in the simulation above is the same as in the Check-forge experiment, except the case when it queries the verification oracle with (r, s) and there is no entry $(\cdot, \cdot, s)$ in the table. However, this case occurs only with negligible probability (see the proof of the Theorem 6 for formal discussion). If the case does not occur, then whenever during the simulation E asks its verification oracle a valid query, D solves the strong DH problem. Hence, if the strong DH is hard relative to G then the YL scheme is unforgeable in the check model. □

4.1 Modified YL scheme

We modify the YL scheme so that its security is based on the strong twin DH problem, which is equivalent to the computational DH problem. Hence, the modified scheme is based on a weaker assumption than the original one.

Modified YL scheme

Let G be a group of prime order q, where G is a subgroup of $\mathbb{Z}_p^*$ for large primes p, q (q is a prime factor of $p-1$). Let g be a generator of G. Define a designated verifier signature scheme as follows:

- *Gen*(1^n) generates a tuple $(x_1, x_2, X_1, X_2, y, Y)$ of private and public keys:
  - A: private key (x_1, x_2), public key (X_1, X_2), where $x_1, x_2 \overset{\$}{\leftarrow} \mathbb{Z}_q$ and $X_1 = g^{x_1}$, $X_2 = g^{x_2}$
  - B: private key $y \overset{\$}{\leftarrow} \mathbb{Z}_q$, public key $Y = g^y$
- *Sig*$(m, \langle x_1, x_2\rangle, Y)$ outputs $\langle r_1, r_2, s\rangle$, where
  - $s = H(m \,||\, t)$, where $t \overset{\$}{\leftarrow} \mathbb{Z}_q^*$
  - $r_1 = (m \,||\, t) \cdot Y^{x_1 s}$
  - $r_2 = (m \,||\, t) \cdot Y^{x_2 s}$
- *Vrf*$(m, \langle r_1, r_2, s\rangle, y, X)$:
  - $m \,||\, t \leftarrow r_1 \cdot X_1^{-ys}$ (parsing)
  - return $(m \,||\, t \overset{?}{=} r_2 \cdot X_2^{-ys} \wedge H(m \,||\, t) \overset{?}{=} s)$
- *Sim*$(m, \langle X_1, X_2\rangle, y)$ outputs $\langle r_1, r_2, s\rangle$, where
  - $s = H(m \,||\, t)$, where $t \overset{\$}{\leftarrow} \mathbb{Z}_q^*$
  - $r_1 = (m \,||\, t) \cdot X_1^{ys}$
  - $r_2 = (m \,||\, t) \cdot X_2^{ys}$

The perfect designated verifier property of the above scheme comes immediately from the fact that

$$Y^{x_1 s} = (g^y)^{x_1 s} = (g^{x_1})^{ys} = X_1^{ys}$$

and

$$Y^{x_2 s} = (g^y)^{x_2 s} = (g^{x_2})^{ys} = X_2^{ys}.$$

Theorem 4. *Let $\mathit{\Pi} = \langle Gen, Sig, Vrf, Sim\rangle$ be the modified YL scheme. Then the algorithm Sim$(\cdot, \langle X_1, X_2\rangle, y)$ produces identically distributed signatures as those produced by the algorithm Sig$(\cdot, \langle x_1, x_2\rangle, Y)$.*

Proof. Let n be some fixed security parameter and $(x_1, x_2, X_1, X_2, y, Y)$ be a tuple of private and public keys generated by $\mathit{Gen}(1^n)$. Fix some r_1, r_2 and s. We need to show, that the following probabilities are equal

$$\Pr\left[m \overset{\$}{\leftarrow} \mathbb{Z}_p^*; (r_1, r_2, s) \leftarrow \mathit{Sig}(m, \langle x_1, x_2\rangle, Y)\right]$$

and

$$\Pr\left[m \overset{\$}{\leftarrow} \mathbb{Z}_p^*; (r_1, r_2, s) \leftarrow \mathit{Sim}(m, \langle X_1, X_2\rangle, y)\right],$$

where the probabilities are taken over random selection of m and random coins of *Sig* and *Sim* respectively.

$$\begin{aligned}
&\Pr\left[m \overset{\$}{\leftarrow} \mathbb{Z}_p^*; (r_1, r_2, s) \leftarrow \mathit{Sig}(m, \langle x_1, x_2\rangle, Y)\right] = \\
&\quad = \Pr\left[m \overset{\$}{\leftarrow} \mathbb{Z}_p^*; t \overset{\$}{\leftarrow} \mathbb{Z}_q^*;\right. \\
&\qquad\qquad \left. s = H(m||t) \wedge r_1 = (m \,||\, t) \cdot Y^{x_1 s} \wedge r_2 = (m \,||\, t) \cdot Y^{x_2 s}\right] \\
&\quad = \Pr\left[m \overset{\$}{\leftarrow} \mathbb{Z}_p^*; t \overset{\$}{\leftarrow} \mathbb{Z}_q^*;\right. \\
&\qquad\qquad \left. s = H(m||t) \wedge r_1 = (m \,||\, t) \cdot X_1^{ys} \wedge r_2 = (m \,||\, t) \cdot X_2^{ys}\right] \\
&\quad = \Pr\left[m \overset{\$}{\leftarrow} \mathbb{Z}_p^*; (r_1, r_2, s) \leftarrow \mathit{Sim}(m, \langle X_1, X_2\rangle, y)\right].
\end{aligned}$$

Hence, the scheme $\mathit{\Pi}$ has the designated verifier property. □

Besides the perfect designated verifier property, the modified YL scheme has also the strong designated verifier property. Let $\mathit{Sim}'(m, \langle X_1, X_2\rangle, Y)$ be an algorithm that outputs a tuple $\langle r_1, r_2, s\rangle$, where

- $s = H(m \,||\, t)$, where $t \overset{\$}{\leftarrow} \mathbb{Z}_q^*$
- $r_1 = (m \,||\, t) \cdot g^{z_1 s}$, where $z_1 \overset{\$}{\leftarrow} \mathbb{Z}_q$
- $r_2 = (m \,||\, t) \cdot g^{z_2 s}$, where $z_2 \overset{\$}{\leftarrow} \mathbb{Z}_q$.

Theorem 5. *Let $\mathit{\Pi} = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim}'\rangle$ denote the modified YL scheme with the Sim' algorithm as above. If the decisional twin DH assumption is hard relative to G, then the scheme $\mathit{\Pi}$ has the strong designated verifier property.*

Proof. Let E be a polynomial adversary and let

$$\varepsilon(n) := \Pr[\mathit{Ind\text{-}SDV}_{E,\mathit{\Pi}}(n) = 1] - \frac{1}{2}.$$

From the adversary E we construct an algorithm D that solves the decisional twin DH problem whenever E wins in the strong designated verifier experiment. Consider the following algorithm D:

Algorithm D:
The algorithm is given $(G, p, q, g, X_1, X_2, Y, Z_1, Z_2)$, where $X_1 = g^{x_1}$, $X_2 = g^{x_2}$, $Y = g^y$ and Z_1, Z_2 are either equal to $g^{x_1 y}, g^{x_2 y}$ or to g^{z_1}, g^{z_2} for $z_1, z_2 \overset{\$}{\leftarrow} \mathbb{Z}_q$.

1. Simulate E in the strong designated verifier experiment with public keys $\langle X_1, X_2\rangle$ and Y. When E asks its oracle a query m, answer a tuple $\langle r_1, r_2, s\rangle$, where:

- $s = H(m \,||\, t)$, where $t \xleftarrow{\$} \mathbb{Z}_q^*$,
- $r_1 = (m \,||\, t) \cdot Z_1^s$,
- $r_2 = (m \,||\, t) \cdot Z_2^s$.

2. At the end of E's execution, it outputs a bit b', output b'.

It is obvious that D runs in a polynomial time. When D's input is a twin DH tuple, i.e. $Z_1 = g^{x_1 y}$ and $Z_2 = g^{x_2 y}$, then D simulates E's oracle exactly as the algorithm $\mathit{Sig}(\cdot, \langle x_1, x_2 \rangle, Y)$. On the other hand, when $Z_1 = g^{z_1}$ and $Z_2 = g^{z_2}$ for $z_1, z_2 \xleftarrow{\$} \mathbb{Z}_q$, then D answers E's queries exactly as $\mathit{Sim}'(\cdot, \langle X_1, X_2 \rangle, Y)$. Let b be a bit chosen randomly in the strong designated verifier experiment. We have:

$$\begin{aligned}
\varepsilon(n) &= \frac{1}{2}\Pr[\mathit{Ind\text{-}SDV}_{E,\Pi}(n) = 1 | b = 1] + \frac{1}{2}\Pr[\mathit{Ind\text{-}SDV}_{E,\Pi}(n) = 1 | b = 0] - \frac{1}{2} \\
&= \frac{1}{2}\Pr[\mathit{Ind\text{-}SDV}_{E,\Pi}(n) = 1 | b = 1] + \frac{1}{2}(1 - \Pr[\mathit{Ind\text{-}SDV}_{E,\Pi}(n) = 0 | b = 0]) - \frac{1}{2} \\
&= \frac{1}{2}\Pr[\mathit{Ind\text{-}SDV}_{E,\Pi}(n) = 1 | b = 1] - \frac{1}{2}\Pr[\mathit{Ind\text{-}SDV}_{E,\Pi}(n) = 0 | b = 0] \\
&= \frac{1}{2}\Pr[D(X_1, X_2, Y, g^{z_1}, g^{z_2}) \to 1] - \frac{1}{2}\Pr[D(X_1, X_2, Y, g^{xy}, g^{xy}) \to 1].
\end{aligned}$$

By the assumption that the decisional DH problem is hard relative to G we conclude that there exists a negligible function negl such that $\varepsilon(n) = \mathrm{negl}(n)$. Hence, the scheme Π has the strong designated verifier property. □

The security of the scheme in the check model can be proved similarly to the proof of the Theorem 3.

Theorem 6. *If the strong twin DH problem is hard relative to G, and H is modeled as a random oracle, then the modified YL scheme is unforgeable in the check model.*

Proof. Let $\Pi = \langle \mathit{Gen}, \mathit{Sig}, \mathit{Vrf}, \mathit{Sim} \rangle$ be the modified YL scheme. Consider a polynomial adversary E attacking Π in the check model. Let

$$\varepsilon(n) := \Pr[\mathit{Check\text{-}forge}_{E,\Pi}(n) = 1].$$

Without loss of generality we can assume that E outputs only those signatures (r_1, r_2, s), which he verified by asking the verification oracle. Let *ValidQuery* denote the event that the adversary E asks its verification oracle a valid query (r_1, r_2, s) (i.e. $\mathit{Vrf}(r_1, r_2, s, y, X) = 1$). Let $Q_H = \{(m_i, t_i, s_i)\}_{i=0}^{q_H(n)}$ be the set of all queries and corresponding answers E asks the random oracle H. Let *Bad* denote the event that E asks a valid query (r_1, r_2, s), but there is no entry $(\cdot, \cdot, s)$ in the set Q_H. We have

$$\begin{aligned}
\varepsilon(n) &\leq \Pr\left[\mathit{ValidQuery}\right] \\
&= \Pr\left[\mathit{ValidQuery} \wedge \mathit{Bad}\right] + \Pr\left[\mathit{ValidQuery} \wedge \overline{\mathit{Bad}}\right] \\
&\leq \Pr\left[\mathit{Bad}\right] + \Pr\left[\mathit{ValidQuery} | \overline{\mathit{Bad}}\right]
\end{aligned} \qquad (2)$$

We claim that $\Pr[\mathit{Bad}]$ is negligible. Intuitively, if H is a random oracle then E is unable to forge a valid query (r_1, r_2, s) without knowing m, t for which $H(m||t) = s$. Formally, let

$Q_V = \{(r_{1,i}, r_{2,i}, s_i)\}_{i=0}^{q_V(n)}$ be the set of all queries made by E to its verification oracle. Let Bad_i denote the event that the i-th query $(r_{1,i}, r_{2,i}, s_i)$ is valid, but there is no previous query to H with response s_i. Obviously, the probability of Bad_i is negligible, since $\mathit{Vrf}(r_{1,i}, r_{2,i}, s_i) = 1$ only if $H(m \,||\, t) = s_i$, where $m \,||\, t \leftarrow r_{1,i} \cdot X_1^{-ys_i}$. If $H(m \,||\, t)$ was not queried previously then it is equal to s_i with probability $1/q$. Hence,

$$\Pr\left[\mathit{Bad}\right] \leq \sum_{i=0}^{q_V(n)} \Pr\left[\mathit{Bad}_i\right] \leq \frac{q_V(n)}{q}. \tag{3}$$

We now show that if the strong twin DH problem is hard, then $\Pr[\mathit{ValidQuery}|\overline{\mathit{Bad}}]$ is negligible. Consider the following algorithm D:

Algorithm D:
The algorithm is given $(G, p, q, g, X_1, X_2, Y)$, where $X_1 = g^{x_1}$, $X_2 = g^{x_2}$, $Y = g^y$ and has oracle access to $\text{2dhp}(X_1, X_2, \cdot, \cdot, \cdot)$.

1. Given E public keys X_1, X_2 and Y. Maintain a hash table of all E's queries to the random oracle. The table consists of tuples (m, t, s) and is initially empty.
2. When E asks its random oracle a query $H(m||t)$, answer in the following way
   - search the table for entry (m, t, s), if such an entry exists, output s,
   - otherwise choose $s \overset{\$}{\leftarrow} \mathbb{Z}_q$, store (m, t, s) in the table and output s.
3. When E asks a query to its verification oracle $\mathit{Vrf}(r_1, r_2, s)$, answer as follows:
   - search the table for entry (m, t, s), if no such entry exists return 0,
   - otherwise compute $Z_1 = (r_1/(m||t))^{1/s} \mod p$ and $Z_2 = (r_2/(m||t))^{1/s} \mod p$ and check, if $\text{2dhp}(X_1, X_2, Y, Z_1, Z_2)$ holds:
     - If so, output (Z_1, Z_2) as a solution of the strong twin DH problem and terminate the execution, otherwise return 0 as an answer to E's verification oracle query.
4. At the end of E's execution, it outputs a tuple (r_1, r_2, s). Check if it is a valid signature as in the previous step. If (r_1, r_2, s) is a valid signature, output Z_1, Z_2, otherwise output $\perp$.

It is obvious that D runs in a polynomial time. Clearly, if Bad does not occur, E's view in the simulation above is the same as in the Check-forge experiment. If E asks a valid query to its verification oracle, then D solves the strong twin DH problem. Since we assume that the strong twin DH problem is hard, we have

$$\Pr[\mathit{ValidQuery}|\overline{\mathit{Bad}}] = \text{negl}(n) \tag{4}$$

By combining equations (2), (3) and (4) we have

$$\begin{aligned} \varepsilon(n) &\leq \Pr\left[\mathit{Bad}\right] + \Pr\left[\mathit{ValidQuery}|\overline{\mathit{Bad}}\right] \\ &\leq \frac{q_V(n)}{q} + \text{negl}(n). \end{aligned}$$

Hence, $\varepsilon(n)$ is negligible. □

By the results of Cash et al. [1], the strong twin DH assumption is equivalent to the computational DH assumption. Thus we can state the following corollary:

Corollary 1. *If the computational DH problem is hard relative to G and H is modeled as a random oracle, then the modified YL scheme is unforgeable in the check model.*

5 Conclusion

We analyzed relationships among different models of unforgeability of designated verifier signature schemes. We proved that the no-message model is equivalent to the check model, where an adversary has oracle access to the verification algorithm. We also presented a separation between the no-message model and the chosen-message model. The separation holds in both standard and strong DV settings.

In the second part of this paper we presented a modification of Yang-Liao [13] scheme. We proved that if the strong twin DH assumption holds the modified scheme has strong designated verifier property and is unforgeable in the check model. These facts together with the results from the Section 3 imply that the modified scheme is unforgeable also in the chosen-message model (under the computational DH assumption).

Acknowledgment. Both authors were supported by VEGA grant No. 1/0266/09.

References

1. D. Cash, E. Kiltz, V. Shoup: The Twin Diffie-Hellman Problem and Application, *Journal of Cryptology*, Vol. 22, No. 4, Springer, 2009.
2. S.S.M. Chow: Identity-based StrongMulti-Designated Verifiers Signatures, *EuroPKI 2006*, LNCS, vol. 4043, pp. 257-259, Springer, 2006.
3. X. Huang, W. Susilo, Y. Mu, F. Zhang: Restricted Universal Designated Verifier Signature, *UIC 2006*, LNCS, vol. 4159, pp. 874-882. Springer, 2006.
4. M. Jakobsson, K. Sako, R. Impagliazzo: Designated Verifier Proofs and Their Applications, *Advances in Cryptology (Proceedings of Eurocrypt '96)*, LNCS vol. 1070, pp. 143–154, Springer, 1996.
5. F. Laguillaumie, D. Vergnaud: Multi-designated Verifiers Signatures, *ICICS 2004*, LNCS, vol. 3269, pp. 495-507, Springer, 2004.
6. J. Lee, J.H. Chang: Comment on Saeedniia et al.'s strong designated verifier signature scheme, *Computer Standards & Interaces*, Vol. 31, pp. 258–260, Elsevier, 2009.
7. Y. Li, W. Susilo, Y. Mu, D. Pei: Designated Verifier Signature: Definition, Framework and New Constructions, *UIC 2007*, LNCS, vol. 4611, pp. 1191–1200, Springer, 2007.
8. H. Lipmaa, G. Wang, F. Bao: Designated Verifier Signature Schemes: Attacks, New Security Notions and A New Construction, *ICALP 2005*, LNCS, vol. 3580, pp. 459-471. Springer, 2005.
9. C.Y. Ng, W. Susilo, Y. Mu: Universal Designated Multi Verifier Signature Schemes, *SNDS 2000*, pp. 305-309, IEEE Press, NJ, New York, 2005.
10. S. Saeednia, S. Kremer, O. Markowitch: An Efficient Strong Designated Verifier Signature Scheme, *Information Security and Cryptology*, LNCS vol. 2971, pp. 40–54, Springer, 2004.
11. R. Steinfeld, L. Bull, H. Wang, J. Pieprzyk: Universal Designated-Verifier Signatures, *ASIACRYPT 2003*, LNCS, vol. 2894, pp. 523-542. Springer, 2003.
12. D. Vergnaud: New Extensions of Pairing-based Signatures into Universal Designated Verifier Signatures, *ICALP 2006*, LNCS, vol. 4052, pp. 58-69. Springer, 2006.
13. F. Yang, C. Liao: A Provably Secure and Efficient Strong Designated Verifier Signature Scheme, *International Journal of Network Security*, Vol. 11, No. 2, pp. 60–64, 2010.
14. R. Zhang, J. Furukawa, H. Imai: Short Signature and Universal Designated Verifier Signature Without Random Oracles, *ACNS 2005*, LNCS, vol. 3531, pp. 483-498, Springer, 2005.