Further Discussions on the Security of a Nominative Signature Scheme

Lifeng Guo[1,2], Guilin Wang[2], and Duncan S. Wong[3]

[1] Institute of Systems Science, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, Graduate School of Chinese Academy of Sciences
Beijing 100080, P.R. China
lfguo@amss.ac.cn

[2] Infocommm Security Department, Institute for Infocomm Research (I^2R)
21 Heng Mui Keng Terrace, Singapore 119613
{stulguo, glwang }@i2r.a-star.edu.sg

[3] Department of Computer Science, City University of Hong Kong
Hong Kong, China
duncan@cityu.edu.hk

Abstract. A nominative signature scheme allows a nominator (or signer) and a nominee (or verifier) to jointly generate and publish a signature in such a way that *only* the nominee can verify the signature and if necessary, *only* the nominee can prove to a third party that the signature is valid. In a recent work, Huang and Wang proposed a new nominative signature scheme which, in addition to the above properties, *only* allows the nominee to convert a nominative signature to a publicly verifiable one. In ACISP 2005, Susilo and Mu presented several algorithms and claimed that these algorithms can be used by the nominator to verify the validity of a published nominative signature, show to a third party that the signature is valid, and also convert the signature to a publicly verifiable one, all *without* any help from the nominee. In this paper, we point out that Susilo and Mu's attacks are actually *incomplete* and *inaccurate*. In particular, we show that there exists no efficient algorithm for a nominator to check the validity of a signature if this signature is generated by the nominator and the nominee *honestly* and the Decisional Diffie-Hellman Problem is hard. On the other hand, we point out that the Huang-Wang scheme is indeed *insecure*, since there is an attack that allows the nominator to generate valid nominative signatures alone and prove the validity of such signatures to a third party.

Keywords: Digital Signature, Nominative Signature.

1 Introduction

Under the assumption that public keys are publicly known, which is normally realized by the public key infrastructure (PKI), a conventional digital signature scheme allows a signer to generate signatures which are *publicly verifiable*. That is, the validity of such a standard signature can be verified by anybody using the signer's public key and the associated message. Digital signature has a wide

range of applications on the Internet and the electronic world at large. It also has quite a number of variants which have been proposed for fulfilling different needs requested by various kinds of applications. Variants include blind signature [3], undeniable signature [5], proxy signature [10], and many others, not to mention schemes involving multiple signers.

In 1996, Kim, Park and Won [8] introduced the notion of *nominative signatures* and proposed the first scheme of this type. Here, we call their scheme the KPW nominative signature scheme. A nominative signature scheme allows a nominator (or signer) and a nominee (or verifier) to work jointly to generate and publish a signature onto some public domain. A published nominative signature is different from a conventional digital signature. *Only* the nominee, using his own private key, can verify the validity of a published nominative signature. Furthermore, if necessary, *only* the nominee can prove (in zero-knowledge) to a third party that the signature is issued to him and is valid. In 2004, Huang and Wang [6] extended the notion of nominative signature in a way that, in addition to the above properties, *only* the nominee can convert a nominative signature to a publicly verifiable one. With this additional property, a scheme of this type is called a *convertible* nominative signature scheme. As mentioned in [8, 6], nominative signatures might be useful in the scenarios where a signed message is personally or commercially sensitive, such as a tax bill, a medical examination report, an ID certificate, etc.

There are several other types of signature schemes that are closely related to nominative signatures, such as (convertible) undeniable signatures [5, 1], designated confirmer signatures [4], and designated verifier signatures [7, 9]. What they are in common is that all of these schemes are trying to prohibit public signature verification. Instead, they prefer putting some restrictions on who or how a signature can be verified, depending on the different needs of their target applications. Specifically, a verifier can check the validity of an undeniable signature only with the help of the signer, while a convertible undeniable signature can further be transformed to a publicly verifiable one only by the signer. For a designated confirmer signature, a verifier can carry out signature verification only with the help of a designated third party (or the signer). In a designated verifier signature scheme, only the designated verifier can be convinced that a signature is issued by the signer, since the designated verifier can simulate such signatures too. Comparing with these schemes, a nominative signature scheme hands over the control of signature verification to the verifier. So, we can consider nominative signatures as the complement of undeniable signatures.

In ACISP 2004, Huang and Wang [6] pointed out that the KPW nominative signature scheme is *not* nominative, since the nominator can also verify and prove the validity of a signature to a third party. They further proposed a modified version in their paper. In ACISP 2005, Susilo and Mu [12] claimed that the Huang-Wang scheme is not nominative either. Specifically, they described several algorithms and claimed that these algorithms can be used by the nominator to (a) verify the validity of a nominative signature, (b) show to a third party that

the signature is valid, and (c) convert the signature to a publicly verifiable one, all *without* any help from the nominee.

But in this paper, we point out that Susilo and Mu's attacks are *incomplete* and *inaccurate*. On the one hands, we show that even a published nominative signature is invalid, Susilo-Mu's algorithms will still enable the nominator to accept it as a valid one. We also show that the nominator cannot prove the validity of a signature to a third party nor convert a nominative signature to a publicly verifiable one using their algorithms. In particular, we show that there exists no efficient algorithm for the nominator to check the validity of a published nominative signature if the signature is generated *honestly* by the nominator in the Signature Generation Phase and the Decisional Diffie-Hellman Problem is hard. We say a signature is generated honestly if the nominator behaves exactly according to the description of the Signature Generation Phase. On the other hand, however, we point out that the Huang-Wang scheme is indeed *insecure*, since there is an attack that allows the nominator to generate valid nominative signatures alone and prove the validity of such signatures to a third party.

The rest of the paper is organized as follows. In Sec. 2, we briefly introduce the components of a nominative signature scheme and review the Huang-Wang nominative signature scheme. In Sec. 3, we review the Susilo-Mu attacks described in [12] that have been claimed to be able to break the Huang-Wang nominative signature scheme. In Sec. 4, we show that Susilo-Mu's algorithms cannot be used to break the Huang-Wang scheme and also show that there exists no efficient algorithm for a nominator to check the validity of a signature if the signature is generated jointly by an honest nominator and some arbitrary nominee. Finally, we conclude the paper in Sec. 5 with some remarks and open problems.

2 Huang-Wang Nominative Signature Scheme

In [6], Huang and Wang proposed a nominative signature scheme which consists of

1. a Signature Generation Phase,
2. a Nominee-only Verification Phase and
3. a Proof of Signature Phase.

In the Signature Generation Phase, the nominator and the nominee interact with each other. At the end of the phase, a signature is generated and published. The signature can only be verified by the nominee during the Nominee-only Verification Phase. In the Proof of Signature Phase, the nominee provides a zero-knowledge proof to a third party that the published signature is issued to him and is valid, that is, the signature is indeed generated by the nominator[4]. The third party is assumed to have only the public information which includes the public keys of both the nominator and the nominee.

[4] To be precise, it should be read as "the signature is indeed generated with the cooperation of the nominator".

Besides these three phases, the Huang-Wang scheme also has two additional phases. They are

4. Signature Conversion Phase and
5. the Universal Verification Phase.

In the Signature Conversion Phase, the nominee converts the published signature to a publicly verifiable signature. After the conversion, that is in the Universal Verification Phase, everyone can check the validity of the signature using the public key of the nominator. Including the last two phases, the nominative signature scheme is called a *convertible* nominative signature scheme [6].

For the formal definitions of nominative signatures, please refer to [6]. In the following, we only review the five concrete phases of the Huang-Wang scheme.

0. Preliminaries: Let p, q be large prime such that $q|p–1$. Let g be an element in $\mathbb{Z}_p^*$ of order q. Assume that the discrete logarithm problem in the group of $\langle g \rangle$ is hard. By $x \in_R X$, we mean picking an element x randomly from X. The public/private key pair of the nominator S is (y_S, x_S) where $x_S \in_R \mathbb{Z}_q$ and $y_S = g^{x_S} \bmod p$. The public/private key pair of the nominee V is (y_V, x_V) where $x_V \in_R \mathbb{Z}_q$ and $y_V = g^{x_V} \bmod p$. In the rest of the paper, we assume that all public keys are publicly known. Let $H : \{0,1\}^* \to \mathbb{Z}_q$ be a one-way hash function. Let $\|$ denote the concatenation between strings. An algorithm $\mathcal{A}$ is called an *efficient* algorithm if $\mathcal{A}$ is a probabilistic polynomial-time algorithm in some security parameter. In the context of our discussions, the security parameter is always considered to be the length of p in binary representation.

1. Signature Generation Phase:

(1) The nominee V randomly picks $R_1, R_2 \in_R \mathbb{Z}_q^*$, computes

$$a = g^{R_1} \bmod p \quad \text{and} \quad c = y_V^{R_2} \bmod p$$

and sends (a, c) to the nominator S.

(2) S randomly chooses $r \in_R \mathbb{Z}_q$, computes

$$\begin{aligned} b &= ag^{-r} \bmod p \\ e &= H(y_V \| b \| c \| m) \\ s' &= r - x_S \cdot e \bmod q \end{aligned}$$

and sends (e, b, s') to V.

(3) V checks if the following equations hold.

$$e \stackrel{?}{=} H(y_V \| b \| c \| m) \quad \text{and} \quad a \stackrel{?}{=} g^{s'} y_S^e b \bmod p$$

If both of them hold, V computes

$$s = s' + R_2 - R_1 \bmod q \tag{1}$$

and outputs a nominative signature $\sigma = (b, c, s)$. Otherwise, output "False".

2. Nominee-only Verification Phase: Given a signature $\sigma = (b, c, s)$ and a message m, the nominee V computes $e = H(y_V \| b \| c \| m)$ and checks whether

$$(g^s y_S^e b)^{x_V} \stackrel{?}{=} c \bmod p$$

holds with equality. If so, the signature is accepted. Otherwise, it is rejected.

3. Proof of Signature Phase: For a published nominative signature $\sigma = (b, c, s)$ for message m, let $e = H(y_V||b||c||m)$ and $d = g^s y_S^e b \bmod p$. The nominee V can confirm or disavow the validity of σ via proving the equality/non-equality of discrete logarithm $\log_d c = \log_g y_V$ or $\log_d c \neq \log_g y_V$ using the interactive zero-knowledge protocol of Michels and Stadler [11]. We refer readers to [6] for the detail on how this proof is conducted interactively.

4. Signature Conversion Phase: To convert a nominative signature $\sigma = (b, c, s)$ into a universally (or publicly) verifiable one, the nominee V just needs to release σ and a non-interactive proof r_m that shows $\log_d c = \log_g y_V$, where $d = g^s y_S^e b \bmod p$ and $e = H(y_V||b||c||m)$.

5. Universal Verification Phase: Anybody can verify the validity of (σ, r_m) by checking whether r_m is a correct non-interactive proof for $\log_d c = \log_g y_V$, where $d = g^s y_S^e b \bmod p$ and $e = H(y_V||b||c||m)$.

3 Susilo-Mu's Attacks

A secure nominative signature scheme should only allow the nominee to verify a published nominative signature during the Nominee-only Verification Phase. In addition, it should only allow the nominee to prove to a third party on the validity of the nominator's signature during the Proof of Signature Phase. To satisfy these two security requirements, the following two conditions should also be satisfied.

1. In the Nominee-only Verification Phase, *the nominator cannot verify a published nominative signature.*
2. In the Proof of Signature Phase, *the nominator cannot provide a proof on the validity of a published nominative signature.*

In [12], Susilo and Mu described several algorithms and claimed that these algorithms can be used by the nominator to compromise these two conditions. They claimed that in the Huang-Wang scheme, the nominator *can* also verify a nominative signature. Below is their algorithm for the nominator to conduct signature verification on any published nominative signature.

> **Susilo-Mu's Verification Algorithm for the Nominator:**
> *For a published nominative signature $\sigma = (b, c, s)$ and a message m, suppose the nominator S stores the transcript of the Signature Generation Phase of σ. Hence S knows the corresponding values of a and s'. Based*

on these information, S computes $e = H(y_V \| b \| c \| m)$ and accepts the signature if the following equation holds with equality:

$$g^s g^{-s'} a \stackrel{?}{\equiv} g^s y_S^e b \bmod p. \tag{2}$$

Note that the verification mechanism above always accepts as long as σ is valid, that is, the nominee accepts σ on message m in the Nominee-only Verification Phase reviewed in the previous section. This can easily be seen from the following facts. From Eq. (1), we have

$$g^s g^{-s'} a = g^{R_2} \bmod p.$$

Also, we have

$$g^s y_S^e b = g^{R_2} \bmod p.$$

Based on this observation, Susilo and Mu [12] concluded that the nominator S can verify a published nominative signature without any collaboration from the nominee V, and therefore, Huang-Wang's scheme does not satisfy the first condition stated in the beginning of this section.

Furthermore, they also claimed that by publishing the value of g^{R_2} and a signature of knowledge [2] by setting g^{R_2} as the public key and σ as the message, the nominator S, without any help from V, can also convert the nominative signature to a publicly verifiable one. Therefore, they claimed that the second condition stated in the beginning of this section cannot be true either. More specifically, they described the following algorithm.

Susilo-Mu's Conversion Algorithm for the Nominator:
To convert a nominative signature $\sigma = (b, c, s)$ to a universally verifiable one, the nominator S reveals σ with a signature of knowledge [2]

$$SPK\{R_2 : d = g^{R_2}\}(\sigma), \quad \text{where} \quad d = g^s g^{-s'} a \bmod p. \tag{3}$$

4 The Invalidity of Susilo-Mu's Attacks

In this section, we show that both of Susilo-Mu's attacks reviewed in previous section are actually invalid. That is, neither of Theorems 1 and 2 in [12] is correct.

First of all, we notice that the nominator S *cannot* generate the signature of knowledge with g^{R_2} as the public key (Eq. (3)). This is because the nominator cannot obtain the value of the corresponding 'private key' R_2, since it is a random number selected by the nominee V. Hence Susilo-Mu's Conversion Algorithm for the Nominator is invalid, contrary to their claim that in the Huang-Wang scheme the nominator can convert a nominative signature to universally verifiable one.

Now, we point out that Susilo-Mu's Verification Algorithm for the Nominator is also invalid. Specifically, we show that there exists a number of invalid signatures $\tilde{\sigma}$ which can be clearly identified to be invalid by the nominee in the Nominee-only Verification Phase but will still be determined as valid ones by

the nominator using the Susilo-Mu verification mechanism reviewed in the previous section. In fact, we will see that there are a large number of such invalid signatures and these invalid signatures can easily be generated by the following invalid signature generation algorithm.

Invalid Signature Generation Algorithm:
Given a valid signature $\sigma = (b, c, s)$ on a message m, that is $e = H(y_V \| b \| c \| m)$ and $(g^s y_S^e b)^{x_V} = c \bmod p$. Randomly pick $\tilde{s} \in_R \mathbb{Z}_q \backslash \{s\}$ and set the output (i.e. the invalid signature) to be $\tilde{\sigma} = (b, c, \tilde{s})$ with the same message m.

Obviously, the nominee will reject the invalid signature $\tilde{\sigma}$ since $\tilde{s} \neq s$. In the following, we show that the nominator will still accept $\tilde{\sigma}$ as a valid signature.

Let $\tilde{r} = \tilde{s} - s \bmod q$. Note that $\tilde{r} \neq 0$. The left-hand-side of Eq. (2) becomes

$$g^{\tilde{s}} g^{-s'} a = g^{\tilde{r}} g^s g^{-s'} a = g^{\tilde{r}} g^{R_2} \bmod p.$$

The right-hand-side of Eq. 2 becomes

$$g^{\tilde{s}} y_S^e b = g^{\tilde{r}} g^s g^e a = g^{\tilde{r}} g^{R_2} \bmod p.$$

Hence Eq. (2) still holds with equality. The reason becomes clear when we remove the common factor g^s from both sides of Eq. (2). Without the term g^s, we can see that the nominator's verification mechanism only checks if $g^{-s'} a \stackrel{?}{=} y_S^e b \bmod p$. This equality can be maintained as long as the message m and the first two components b and c of the original (valid) signature remain unchanged. As a result, Susilo and Mu's proposed verification mechanism for the nominator in the Nominee-only Verification Phase is invalid.

Note that such invalid signatures could be revealed by the nominee V, the nominator S who knows (b, c) and m, and any third party who accepts $\sigma = (b, c, s)$ as a valid nominative signature for message m after an interactive or non-interactive confirmation with the nominee V.

In the following theorem, we further show that if the nominator is *honest* in the entire Signature Generation Phase and does not know the private key of the nominee, the nominator cannot determine whether an *alleged* signature is valid or not in the Nominee-only Verification Phase under the assumption that the Decisional Diffie-Hellman Problem (DDH) is hard. By *honest* nominator, we mean that the nominator follows exactly the protocol described in Sec. 2 for the entire Signature Generation Phase. In other words, when the nominator S receives a pair $(a, c) \in \mathbb{Z}_p^* \times \mathbb{Z}_p^*$, S sends back a triple (e, b, s') such that $b = ag^{-r} \bmod p$, $e = H(y_V \| b \| c \| m)$ and $s' = r - x_S e \bmod q$ where $r \in_R \mathbb{Z}_q$. A signature $\sigma^* = (b^*, c^*, s^*)$ is called *alleged* signature if $(b^*, c^*, s^*) \in \mathbb{Z}_p^* \times \mathbb{Z}_p^* \times \mathbb{Z}_q$.

Theorem 1. *Let S^{honest} be an efficient honest nominator and V^* be an arbitrary and efficient nominee, that is, V^* does not necessarily follow the behave of V described in the Signature Generation Phase in Sec. 2. For any alleged signature*

generated by S^{honest} and V^* in the Signature Generation Phase, if S^{honest} can determine the validity of the signature in the Nominee-only Verification Phase with non-negligible success rate, then there exists an efficient algorithm that can solve the DDH problem with non-negligible success rate.

Proof. We construct an algorithm $\mathcal{D}$ which runs S^{honest} for solving the DDH problem. Given a random DDH problem instance $(g, g_1, g_2 = g^u \bmod p, g_3 = g_1^v \bmod p)$ in which there is half chance that $u = v$ and the other half chance that $u \neq v$. Below is the description of $\mathcal{D}$ with input (g, g_1, g_2, g_3).

1. Set the public/private key pair (y_S, x_S) of the honest nominator S^{honest} to $y_S = g^{x_S} \bmod p$ and $x_S \in_R \mathbb{Z}_q$.
2. Set the public key of the nominee V^* to $y_V = g_1$.
3. $\mathcal{D}$ simulates V^* in the Signature Generation Phase and interacts with S^{honest} as follows.
   (a) V^* randomly picks $R \in_R \mathbb{Z}_q$.
   (b) V^* sends (a, c) and a message $m \in \{0,1\}^*$ to S^{honest} where $a = g_2 g^R \bmod p$ and $c = g_3$. Technically, S^{honest} is invoked with input (a, c).
   (c) S^{honest} sends back (e, b, s') such that $e = H(y_V \| b \| c \| m)$, $s' = r - x_S e \bmod q$ and $b = a g^{-r} \bmod p$ for some random element r of $\mathbb{Z}_q$. Since S^{honest} is an honest nominator, the equation $g^{s'} y_S^e b = a \bmod p$ must hold.
   (d) V^* generates an alleged signature $\sigma^* = (b^*, c^*, s^*)$ by setting $b^* = b$, $c^* = c$ and $s^* = s' - R \bmod q$. This completes the Signature Generation Phase.
4. $\mathcal{D}$ simulates the Nominee-only Verification Phase and invokes S^{honest} with input (σ^*, m). S^{honest} is to determine whether the alleged signature σ^* is valid or not.
5. If S^{honest} returns that σ^* is valid, then $\mathcal{D}$ concludes that (g, g_1, g_2, g_3) is a DDH tuple, that is, $u = v$. Otherwise, $\mathcal{D}$ concludes that it is not a DDH tuple, that is, $u \neq v$.

Obviously, the running time of $\mathcal{D}$ is in polynomial of that of S^{honest}. Also the simulation is perfect, that is, the simulated environment is computationally indistinguishable from a real environment from S^{honest}'s point of view. We now show that the alleged signature σ^* is valid *if and only if* $(g, g_1, g_2 = g^u, g_3 = g_1^v)$ is a DDH tuple, that is, $u = v$.

On the one hand, note that if $\sigma^* = (b^*, c^*, s^*)$ is valid, then

$$(g^{s^*} y_S^e b^*)^{x_V} = c^* \bmod p, \tag{4}$$

where $e = H(y_V \| b^* \| c^* \| m)$ and $x_V = \log_g g_1$ which is the discrete logarithm of g_1 to the base g. Since $g^{x_V} = g_1 \bmod p$, $y_S^{x_V} = g_1^{x_S} \bmod p$ and $(b^*)^{x_V} = (g_2 g^R)^{x_V} g_1^{-r} = g_1^u g_1^R g_1^{-r} \bmod p$, Eq.(4) can be rewritten as

$$g_1^{s^*} g_1^{x_S e} g_1^u g_1^R g_1^{-r} = g_1^v \bmod p.$$

That is,

$$s^* + x_S e + u + R - r = v \bmod q.$$

Since $s^* = s' - R = r - x_S e - R \bmod q$, we have $u = v$.

On the other hand, if (g, g_1, g_2, g_3) is not a DDH tuple, based on the similar derivation, we can see that Eq.(4) would not hold. Therefore, if S^{honest} can determine the validity of any alleged signature non-negligibly, $\mathcal{D}$ can solve the DDH problem with non-negligible success rate. This completes the proof. □

This theorem not only indicates that Susilo-Mu's verification mechanism cannot help the nominator determine the validity of a published nominative signature, but also shows that there is no efficient algorithm to do so unless DDH problem can be solved efficiently. This statement is true even the nominator keeps all the transcripts of all the runs of the Signature Generation Phase, that is, r and s' are not deleted by the nominator after each run of the Signature Generation Phase.

Remark 1. Actually, Theorem 1 can easily be strengthened as "The (honest) nominator can determine the validity of an alleged signature *if and only if* he can solve the DDH problem". The reason is almost obvious: If the DDH problem is solvable, then an alleged signature $\sigma^* = (b^*, c^*, s^*)$ can be verified by simply determining whether $(g, y_V, g^{s^*} y_S^e b^* \bmod p, c^*)$ is a DDH tuple, which is equivalent to checking $(g^{s^*} y_S^e b^*)^{x_V} \stackrel{?}{=} c^* \bmod p$. As in the above, here $e = H(y_V||b^*||c^*||m)$.

Remark 2. Note that in the Huang-Wang scheme the nominator can "forge" a nominative signature *alone*, i.e., without the nominee's cooperation. This is very easy, he can do this by first selecting random numbers R_1 and R_2 and then computing values of a and c. After that, a valid signature is generated according to the signing algorithm. For such a "forged" signature, the nominator of course knows its validity and can confirm this fact to a third party since he knows the value of R_2. This is the reason why we emphasize the nominator should be *honest* in Theorem 1. Therefore, the Huang-Wang scheme is indeed insecure, since the nominator can generate a valid nominative signature *alone*, without the nominee's cooperation.

5 Concluding Remarks

In this paper, we pointed out that in the Huang-Wang nominative signature scheme [6], Susilo and Mu's attacking techniques described in [12] cannot help the nominator determine the validity of a published nominative signature, even the nominator keeps the entire transcripts of all the runs of the Signature Generation Phase. We also showed that Susilo-Mu's conversion method, which is intended to convert a nominative signature to a publicly verifiable one, does not work. Moreover, we proved that as long as the nominator is honest during the Signature Generation Phase and does not know the private key of the nominee, not just the Susilo-Mu's attacking algorithm does not work, there is no any efficient algorithm that can help the nominator determine the validity of a published nominative signature during the Nominee-only Verification Phase under the DDH assumption. This statement is true even the nominator keeps all the transcripts of all the runs of the Signature Generation Phase.

On the other side, although we showed that, under the DDH assumption, the nominator cannot determine whether an alleged signature is valid or not under the condition that the signature is generated by an *honest* nominator with the cooperation of a nominee, we also pointed out the insecurity of the Huang-Wang scheme. That is, the nominator can forge valid nominative signatures and prove the validity of such signatures to a third party alone, without the nominee's cooperation. Therefore, in the future work a rigorous and formal security model needs to be presented, and generic constructions with formal proof for nominative signatures are highly desirable.

References

1. J. Boyar, D. Chaum, I. Damgard and T. Pedersen. Convertible Undeniable Signatures. In: *Proc. of Advances in Cryptology - CRYPTO'90*, LNCS 537, pp. 189–208, Springer-Verlag, 1990.
2. J. Camenisch. Efficient and Generalized Group Signatures. In: *Proc. of Advances in Cryptology - Eurocrypt'97*, LNCS 1233, pp. 465-479, Springer-Verlag, 1997.
3. D. Chaum. Blind Signatures for Untraceable Payments. In: *Proc. of Advances in Cryptology-CRYPTO'82*, LNCS 403, pp. 199-203, Springer-Verlag, 1982.
4. D. Chaum. Designated Confirmer Signatures. In: *Proc. of Advances in Cryptology - EUROCRYPT'94*, LNCS 950, pp. 86-91, Springer-Verlag, 1994.
5. D. Chaum and H. Antwerpen. Undeniable Signatures. In: *Proc. of Advances in Cryptology-CRYPTO'89*, LNCS 435, pp. 212-216, Springer-Verlag, 1989.
6. Z. Huang and Y. Wang. Convertible Nominative Signatures. In: *Proc. of Information Security and Privacy (ACISP'04)*, LNCS 3108, pp. 348-357, Springer-Verlag, 2004.
7. M. Jakobsson, K. Sako and R. Impagliazzo. Designated Verifier Proofs and Their Applications. In: *Proc. of Advances in Cryptology - EUROCRYPT'96*, LNCS 1070, pp. 143-154, Springer-Verlag, 1996.
8. S.J. Kim, S.J. Park, and D.H. Won. Zero-Knowledge Nominative Signatures. In: *Proc. of PragoCrypt' 96, International Conference on the Theory and Applications of Cryptology*, pp. 380-392, 1996.
9. H. Lipmaa, G. Wang, and F. Bao. Designated Verifier Signature Schemes: Attacks, New Security Notions and A New Construction. In: *Proc. of the 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, LNCS 3580, pp. 459-471, Springer-Verlag, 2005.
10. M. Mambo, K. Usada and E. Okamoto. Proxy Signatures for Delegating Signing Operation. In: *Proc. of the Third ACM Conference on Computer and Communication Security (CCS'96)*, pp. 48-57, ACM Press, 1996.
11. M. Michels and M. Stadler. Efficient Convertible Undeniable Signature Schemes. In: *Proc. of 4th Annual Workshop on Selected Areas in Cryptography (SAC'97)*, pp. 231-244, 1997.
12. W. Susilo and Y. Mu. On the Security of Nominative Signatures. In: *Proc. of Information Security and Privacy (ACISP'05)*, LNCS 3547, pp. 329-335, Springer-Verlag, 2005.