DESIGNATED-VERIFIER PROXY SIGNATURE SCHEMES

Guilin Wang
Instititue for Infocomm Research (I^2R)
21 Heng Mui Keng Terrace, Singapore 119613
glwang@i2r.a-star.edu.sg

Abstract: In a proxy signature scheme, a user delegates his/her signing capability to another user in such a way that the latter can sign messages on behalf of the former. In this paper, we first propose a provably secure proxy signature scheme, which is based on a two-party Schnorr signature scheme. Then, we extend this basic scheme into designated-verifier proxy signatures (DVPS). More specifically, we get two versions of DVPS: weak DVPS and strong DVPS. In both versions, the validity of a proxy signature can be checked only by the designated verifier. In a weak DVPS scheme, however, the designated verifier can further convert such proxy signatures into public verifiable ones, while a strong DVPS scheme does not have the same property even if the designated verifier's secret key is revealed willingly or unwillingly. In addition, we briefly discuss some potential applications for DVPS.

Keywords: proxy signature, digital signature, information security.

1. INTRODUCTION

Proxy Signatures. In a proxy signature scheme, one user Alice, called *original signer*, delegates her signing capability to another user Bob, called *proxy signer*. After that, the proxy signer Bob can sign messages on behalf of the original signer Alice. Upon receiving a proxy signature on some message, a verifier can validate its correctness according to a given verification procedure, and further be convinced of the original signer's agreement on the signed message. Proxy signature schemes have been

suggested for use in a number of applications, including electronic commerce, mobile agents, and distributed shared object systems etc [15, 4, 28].

Most existing proxy signature schemes are constructed in the following way. The original signer Alice sends a specific message and the corresponding signature to the proxy signer Bob, who then uses this information to derive a proxy secret key. With this secret key, Bob can produce proxy signatures by employing a specified standard signature scheme. When a proxy signature is given, a verifier first recovers the proxy public key from some public information, and then checks its validity according to the corresponding standard signature verification procedure.

Mambo et al. firstly introduced the concept of proxy signatures and proposed several constructions in [17,18]. After that, a number of new schemes and improvements have been proposed [4, 14-16, 29]; however, most of them do not fully meet the desired security requirements (see Section 2.2). In [14], Kim et al. introduced the concept of partial delegation by warrant, and proposed a threshold proxy signature, in which the original signer's signing ability is shared among a delegated group of n proxy singers such that only t or more of them can generate proxy signatures cooperatively. Lee et al. [15] constructed mobile agents for e-commerce applications from non-designated proxy signature, in which a warrant does not specify the identity of a proxy signer so any possible proxy signer may respond this delegation and become a proxy signer. Furthermore, Lee et al. [16] investigated whether a secure channel for delivery of a signed warrant is necessary in existing schemes. Their results show that if secure channels are not provided, the MUO scheme [17] and the LKK scheme [15] all are insecure. To avoid the usage of secure channels and overcome some other weaknesses, they proposed new improvements. However, Wang et al. [28] showed that all of those schemes and improvements proposed in [15-16] are insecure by demonstrating several kinds of attacks. Boldyreva et al. [4] presented the formal model and security notion for proxy signature, i.e., the existential unforgeablity against adaptive chosen-message attacks [10].

Designated-Verifier Signatures. In 1996, Jakobsson et al. introduced a new primitive called *designated-verifier proofs* [13]. Such proofs enable a prover Alice to convince a designated verifier Bob that a statement is true. However, Bob cannot use such proofs to convince a third party of this fact. The reason is that Bob himself can simulate such proofs. Here is their basic idea. When Alice wants to convince only the designated verifier Bob a statement Θ, she actually proves the statement "Θ *is true* or *I knows Bob's secret key*". Upon receiving such a proof, Bob is convinced that the statement Θ must be true, since he knows that this proof is not generated by

himself and that his secret key is not compromised. However, a third party cannot accept the statement Θ from such a proof since this proof may be generated by Bob even if Θ is false. Furthermore, Jakobsson et al. proposed an elegant non-interactive designated-verifier proof for Chaum's zero-knowledge undeniable signature scheme [7] to avoid blackmailing [9, 12] and mafia attacks [8]. In other words, they introduced a *designated-verifier signature scheme* in the sense that only the designated verifier can be convinced that a signature is issued by the claimed signer. However, Wang [27] pointed out this scheme is insecure since a dishonest signer can cheat a designated verifier easily.

Note that in Jakobsson et al.'s scheme any verifier can validate a signature though he does not know whether this signature is produced by the signer or simulated by the designated verifier. In [22], however, Saeednia et al. recently proposed a *strong designated-verifier signature scheme* in the sense that without the knowledge of the designated verifier's secret key, any third party cannot check the validity of such signatures. Compared with Jakobsson et al.'s scheme, their scheme is very efficient in both respects of communications and computation. In addition, Steinfeld et al. introduced a new type of signature scheme called *universal designated-verifier signature* (UDVS) [25, 26]. Such a scheme enables *any* holder of a signature (not necessarily the signer) to designate the signature to a third party as the designated-verifier.

Our Work. As mentioned above, Wang et al. [28] demonstrated several attacks on several DLP-based proxy signature schemes. Those attacks mainly result from the fact that a valid proxy key pair can be forged by an adversary, including the original signer and the proxy signer. However, Wang et al. did not provide improvements to avoid such attacks. In this paper, we first propose a new proxy signature scheme, which is based on the two-party Schnorr signature scheme proposed by Nicolosi et al. [20]. The new scheme is provably secure and as efficient as the schemes in [15-16,4].

Then, by combining the ideas of proxy signatures and designated-verifier signatures, we extend this basic scheme to designated-verifier proxy signatures (DVPS for short). More specifically, we get two versions of DVPS: weak DVPS and strong DVPS. In both versions, the validity of a proxy signature can be checked only by the designated proxy signer. In a weak DVPS scheme, however, the designated verifier can further convert such proxy signatures into public verifiable ones, while a strong DVPS scheme does not have the same property even if the designated verifier's secret key is revealed willingly or unwillingly.

In addition, we briefly discuss some potential applications for DVPS in electronic commerce settings.

Structure. The rest of this paper is organized as follows. Section 2 introduces the computational assumptions, security requirements for proxy signature schemes, and notations. We introduce the new (basic) proxy signature scheme in Section 3, and then extend this scheme into designated-verifier proxy signatures (DVPS) in Sections 4. Finally, Section 5 concludes the paper and points out future work.

2. PRELIMINARIES

2.1 Assumptions

We review the following computational assumptions that are related to the security of our proxy signature schemes constructed in this paper.

Assumption 1: Discrete Logarithm (DL) assumption. *Let G_q=<g> be a cyclic multiplicative group generated by g of order q. Then, on inputs $(g, g^x) \in G_q^2$ where $x \in Z_q$ is a random (unknown) number, there is no probabilistic polynomial-time (PPT) algorithm that outputs the value of x with non-negligible probability.*

Assumption 2: Computational Diffie-Hellman (CDH) assumption. *Let G_q=<g> be a cyclic multiplicative group generated by g of order q. Then, on inputs $(g, g^x, g^y) \in G_q^3$ where $x, y \in Z_q$ are random (unknown) numbers, there is no PPT algorithm that outputs the value of g^{xy} with non-negligible probability.*

Assumption 3: Decisional Diffie-Hellman (DDH) assumption. *Let G_q=<g> be a cyclic multiplicative group generated by g of order q. Then, on inputs $(g, g^x, g^y, g^z) \in G_q^4$ where $x, y, z \in Z_q$ are random (unknown) numbers, there is no PPT algorithm that distinguishes with non-negligible probability whether g^{xy} and g^z are equal.*

Those computational assumptions are widely believed to be true for many cyclic groups, such as the multiplicative subgroup G_q=<g> of the finite field Z_p, where p is a large prime and q is a prime factor of p-1. In practice, |p|=1024 and |q|=160 are considered to be suitable for most current security applications. More discussions on those assumptions can be found in [5, 2].

2.2 Definitions

Definition 1. A *proxy signature scheme* is usually comprised of the following procedures:

- Setup: On input of a security parameter l, this probabilistic algorithm outputs two secret/public key pairs (x_A, y_A) and (x_B, y_B) for the original signer Alice and the proxy signer Bob. Note that those key pairs may be used in a standard signature scheme at the same time.
- Proxy Key Pair Generation: The original signer Alice and the proxy signer Bob execute this interactive randomized algorithm to generate a proxy key pair (x_P, y_P) for Bob, such that only Bob knows the value of x_P, while y_P is public or publicly recoverable.
- Proxy Signature Generation: The proxy signer Bob runs this (possibly probabilistic) algorithm to generate a proxy signature σ for a message m by using the proxy secret key x_P.
- Proxy Signature Verification: A verifier runs this deterministic algorithm to check whether an alleged proxy signature σ for a message m is valid with respect to a specific original signer and a proxy signer.

The security requirements for proxy signature are first specified in [17,18], and later are kept almost the same besides being enhanced in [15], and formalized in [4].

Definition 2. A *secure* proxy signature scheme should satisfy the following requirements:

- **Verifiability**: From the proxy signature, a verifier can be convinced of the original signer's agreement on the signed message.
- **Identifiability**: Anyone can determine the identities of the corresponding original signer and proxy signer from a proxy signature.
- **Unforgeability**: Only the designated proxy signer can create a valid proxy signature on behalf of the original signer. In other words, the original signer and other third parties who are not designated as proxy signers cannot create a valid proxy signature.
- **Undeniability**: Once a proxy signer creates a valid proxy signature on behalf of an original signer, he cannot repudiate the signature creation against anyone else.
- **Prevention of misuse**: The proxy signer cannot use the proxy secret key for purposes other than generating valid proxy signatures. In case of misuse, the responsibility of the proxy signer should be determined explicitly.

2.3 Notations

Throughout this paper, p and q are two large primes such that $q|(p\text{-}1)$ and $G_q=<g>$ is a q-order multiplicative subgroup of Z_p^* generated by an element $g \in Z_p^*$. The discrete logarithm problem in G_q is assumed to be difficult. Hereafter, we call three such integers (p, q, g) a *DLP-triple*. Let $h(\,)$ and $h'(\,)$ be two secure cryptographic hash functions. In addition, we suppose that the original signer Alice and the proxy signer Bob possess *certified key pairs* $(x_A, y_A = g^{x_A} \bmod p)$ and $(x_B, y_B = g^{x_B} \bmod p)$, respectively. Here, a certified key pair (x_A, y_A) means that Alice knows the private key x_A and has to prove her knowledge of x_A when she registers her public key certificate with a certificate authority (CA). Actually, this is a recommended practice for issuing public key certificates [1,19], and can be used to prevent rogue-key attacks [4]. In addition, we denote by m_w the *warrant* which specifies the delegation period, what kind of message m is delegated, and the identities of the original signer and the proxy signer, etc.

3. BASIC PROXY SIGNATURE SCHEME

In this section, we propose a new proxy signature scheme. The basic idea is that the provably secure two-party Schnorr signature scheme proposed in [20] is used to generate a proxy key pair (x_P, y_P) such that

$$g^{x_P} = y_P = (y_A \cdot y_B)^{h(m_w, r_P)} \cdot r_P \bmod p), \tag{1}$$

where r_P is a public value, and m_w is a warrant which specifies the related information about a proxy delegation. In fact, (r_P, x_P) is exactly a two-party Schnorr signature on message m_w. The point is that (a) *only* Bob knows the value of x_P, and (b) a valid tuple (r_P, x_P) can *only* be generated by Alice and Bob *jointly*. Therefore, x_P can be used as the proxy secret key to generate proxy signatures according to a standard DLP-based signature scheme. At the same time, a verifier can validate such proxy signatures after recovering the public proxy key y_P from Eq. (1).

In the following description of our scheme, it is assumed that Alice and Bob have agreed on a warrant m_w before generating a proxy key pair for Bob. In addition, as pointed in [20], the hash function $h'(\,)$ can be replaced by any secure commitment scheme.

Proxy Key Generation. To generate a proxy key pair (x_P, y_P) for the proxy signer Bob, Alice and Bob execute the following interactive protocol jointly.

(1) Alice picks a random number $k_A \in Z_q^*$, computes $r_A = g^{k_A} \bmod p$ and $c=h'(r_A)$, and then sends c to Bob.

(2) Similarly, Bob first chooses a random number $k_B \in Z_q^*$, then computes $r_B = g^{k_B} \bmod p$ and replies Alice with (c, r_B).

(3) When (c, r_B) is received, Alice checks whether $r_B^q \equiv 1 \bmod p$. If this is true, she computes $r_P = r_A \cdot r_B \bmod p$, $s_A = k_A + x_A \cdot h(m_w, r_P) \bmod q$, and sends the pair (r_A, s_A) to Bob.

(4) Upon receiving (r_A, s_A), Bob computes $r_P = r_A \cdot r_B \bmod p$, and then checks whether $r_A^q \equiv 1 \bmod p$, $c \equiv h'(r_A)$, and $g^{s_A} \equiv y_A^{h(m_w, r_P)} \cdot r_P \bmod p$. If all validations pass, he calculates $s_B = k_B + x_B \cdot h(m_w, r_P) \bmod q$, and finally sets his proxy key pair (x_P, y_P) by

$$x_P = s_A + s_B \bmod q, \quad \text{and} \quad y_P = g^{x_P} \bmod p. \tag{2}$$

It is easy to know that the above defined proxy key pair (x_P, y_P) satisfies Eq. (1), i.e., (r_P, x_P) is a standard Schnorr signature [23] on the warrant m_w with respect to the public key $y_A y_B \bmod p$.

In addition, note that in the above proxy key generation procedure, we do not assume the communication channel between Alice and Bob is secure. Namely, public channel could be used unless delegation privacy is required. The reason is that the exchanged data, i.e., m_w, r_A, s_A, r_B, s_B etc., are useless for other party (to forge proxy key pairs or proxy signatures).

Proxy Signature Generation. To generate a proxy signature on a message m that conforms to the warrant m_w, the proxy signer Bob performs the same operations as in the standard Schnorr signature scheme [23]. That is, he first selects a random number $k \in Z_q^*$, then computes $r = g^k \bmod p$ and $s = k + x_P \cdot h(m, m_w, r) \bmod q$. The resulting proxy signature on message m is $\sigma = (m_w, r_P, r, s)$.

Proxy Signature Verification. To verify the validity of an alleged proxy signature σ for message m, a verifier operates as follows:

(1) Check whether the message m conforms to the warrant m_w. If not, stop. Otherwise, continue.

(2) Check whether Alice and Bob are specified as the original signer and the proxy signer in the warrant m_w, respectively.

(3) Recover the proxy public key y_P from public information by computing $y_P = (y_A \cdot y_B)^{h(m_w, r_P)} \cdot r_P \bmod p$.

(4) Accept the proxy signature σ *if and only if* the following equality holds:

$$g^s = y_P^{h(m,m_w,r)} \cdot r \bmod p. \quad (3)$$

In the above proxy scheme, when the proxy singer Bob generates a proxy signature the warrant m_w is embedded in the input of the hash function $h(\ )$. The aim is to use m_w as an identifier of proxy signatures.

We now discuss the security of our above scheme. According to the results in [20] and [21], we have the following Proposition 1 and 2. Then, Proposition 3 holds.

Proposition 1 (Theorems 1 and 2 of [20]). *In the random oracle model, if an adversary, who may compromise the original signer or the proxy signer (but not both), can forge a proxy key pair (x_P, y_P) that satisfies Eq. (1) with respect to a pair (m_w, r_P) in probabilistic polynomial time (PPT) with non-negligible probability, then the discrete log problem in the multiplicative subgroup <g> can be solved in PPT with non-negligible probability.*

Proposition 2 [21]. *Under the assumption that the discrete log problem in the multiplicative subgroup <g> is intractable, the Schnorr signature scheme is secure in the random oracle model.*

Proposition 3. *Under the assumption that the discrete log problem in the multiplicative subgroup <g> is intractable, the proposed proxy signature scheme is secure in the random oracle model.*

Proof (*Sketch*): In our scheme, we use Nicolosi et al.'s *provably secure* two-party Schnorr signature scheme [20] to generate proxy key pair (x_P, y_P). That is, in their scheme a two-party Schnorr signature for a message can *only* be generated by the two related parties jointly. In our scheme, a valid proxy key pair (x_P, y_P) (defined by Eq. (1)) implies that (r_P, x_P) is exactly Alice and Bob's valid two-party Schnorr signature on the warrant m_w in Nicolosi et al.'s scheme. Therefore, anybody (including Alice and Bob) cannot generate a valid proxy key pair independently. Meanwhile, without a valid proxy key pair anybody cannot generate a proxy signature such that Eq. (3) is satisfied. Because the proxy signature generation algorithm is just the Schnorr scheme [23], which is also provably secure [21] in the random oracle model [3]. Therefore, we conclude that our proxy scheme is unforgeable. Other security requirements are also met in our new scheme, since we can provide similar security analysis as done in [15-17].

4. DESIGNATED-VERIFIER PROXY SIGNATURES

We now present two designated-verifier proxy signature schemes, in which proxy signatures can be only verified by a designated-verifier. Those DVPS schemes are constructed from the basic proxy signature scheme introduced in Section 3. However, note that the Triple Schnorr proxy signature proposed in [4] could also be used as the basic scheme in a similar way.

In the following description, it is supposed that the original signer Alice and the proxy signer Bob have agreed on a warrant m_w before generating a proxy key pair. In addition, we assume Cindy be the designated verifier with certified key pair $(x_C, y_C = g^{x_C} \bmod p)$. Other system parameters are the same as in previous section. Proxy key generation procedure is the same as our basic scheme described in Section 3. That is, the original signer Alice and the proxy signer Bob jointly generate a proxy key pair (x_P, y_P) for Bob such that the proxy public key y_P can be recovered from Eq. (1), and that only Bob knows the value of the proxy secret key x_P.

4.1 Weak Designated-Verifier Proxy Signature Scheme

Proxy Signature Generation. To generate a weak designated-verifier proxy signature on a message m that conforms to the warrant m_w, the proxy signer Bob performs as follows. He first selects a random number $k \in Z_q^*$ at uniform, then computes (r, r', s) by Eq. (4), and sends the proxy signature $\sigma' = (m_w, r_P, r', s)$ to the designated verifier Cindy.

$$\begin{aligned} r &= g^k \bmod p, \\ r' &= y_C^k \bmod p, \\ s &= k + x_P \cdot h(m, m_w, r) \bmod q. \end{aligned} \tag{4}$$

Proxy Signature Verification. To verify the validity of a weak DVPS σ', the designated verifier Cindy operates as follows:

(1) Check whether the message m conforms to the warrant m_w. If not, stop. Otherwise, continue.
(2) Check whether Alice and Bob are specified as the original signer and the proxy signer in the warrant m_w, respectively.
(3) Recover the values of r and the proxy public key y_P by computing $r = (r')^{x_C^{-1}} \bmod p$ and $y_P = (y_A \cdot y_B)^{h(m_w, r_P)} \cdot r_P \bmod p$.

(4) Accept the proxy signature σ' *if and only if* the following equality holds:

$$g^s = y_P^{h(m,m_w,r)} \cdot r \bmod p. \quad (5)$$

The essence of the above scheme is that to restrict the publicly verifiability of a proxy signature, we simply encrypted the value r by releasing $r' = y_C^k \bmod p$. Therefore, only the designated verifier Cindy can recover r from r' by using her secret key x_C, and then check the validity of such an encrypted proxy signature. Note that in our above scheme, the designated verifier Cindy can convince any third party to accept such a proxy signature σ' by simply releasing $\sigma = (m_w, r_P, r, s)$. Weak designated-verifier proxy signature schemes might be suitable in the settings where both the proxy signer and the designated verifier want that without their help, any third party cannot validate proxy signatures. We now state the security of the above scheme as follows.

Proposition 4. *Under the assumption that the Diffie-Hellman problem in the multiplicative subgroup <g> is intractable, the proposed weak designated-verifier signature scheme is secure in the random oracle model.*

Proof: We first prove that except the designated-verifier Cindy (and the proxy signer Bob), any third party (including the original signer Alice) cannot check the validity of a weak DVPS $\sigma' = (m_w, r_P, r', s)$ for message m. First of all, note that without the value of r the third party cannot check the validity of σ'. In other words, to validate σ' the third party has to recover the value of r from public information. Under the assumptions that DL is difficult and that $h(\ )$ can be modeled as a random function [3], neither (m_w, r_P, y_P, y_C) nor s can be used to reveal x_C or recover r. Consequently, the third party can only use y_C and r' to recover the value of r. That is, on input $(g, y_C = g^{x_C} \bmod p, r' = g^{k \cdot x_C} \bmod p)$, the third party wants to output $r = g^k \bmod p$. In [2], it is proved that this problem (called *Divisible Computation Diffie-Hellman Problem*) is as difficult as the CDH problem. Therefore, under the CDH assumption, only the designated verifier Cindy can check the validity of a week designated-verifier proxy signature.

Now, we prove that the unforgeability. According to Proposition 3, any adversary who is not delegated as a proxy signer by Alice cannot forge a valid proxy key pair (x_P, y_P). Furthermore, we claim that given a proxy public y_P even the designated verifier Cindy, who knows her secret key x_C but does not know the proxy secret key x_P corresponding to y_P, cannot forge a valid proxy signature for a new message m that never appears in Cindy's records of known signature-message pairs. If this is not the fact, i.e., Cindy can forge a valid weak DVPS $\sigma' = (m_w, r_P, r', s)$ for a new message m. Then,

Cindy can compute $r = (r')^{x_C^{-1}} \bmod p$. The latter means that Cindy can forge a Schnorr signature $\bar{\sigma} = (r, s)$ for message (m, m_w) with respect to the public key y_P. This is contrary to the Proposition 2, i.e., the provable security of the Schnorr signature scheme. Therefore, only the delegated proxy signer Bob can generate valid weak designated-verifier proxy signatures.

4.2 Strong Designated-Verifier Proxy Signature Scheme

Based on the basic proxy scheme proposed in Section 3 and Saeednia et al.'s strong designated-verifier signature scheme [22], we now construct a strong designated-verifier proxy signature scheme. In the new scheme, the designated-verifier Cindy can verify that a proxy signature is signed by the proxy signer Bob, but she is unable to convince anyone else of this fact. Because others know that such signatures may be simulated by the designated-verifier Cindy.

Proxy Signature Generation. To generate a strong designated-verifier proxy signature on a message m that conforms to the warrant m_w, Bob performs as follows. He first selects two random numbers $k \in Z_q$ and $t \in Z_q^*$, then computes (r, c, s) by Eq. (6), and sends the proxy signature $\sigma = (m_w, r_P, c, s, t)$ to the designated verifier Cindy.

$$\begin{aligned} r &= y_C^k \bmod p, \\ c &= h(m, m_w, r), \\ s &= kt^{-1} - x_P \cdot c \bmod q. \end{aligned} \tag{6}$$

Proxy Signature Verification. To verify the validity of a strong DVPS σ, the designated-verifier Cindy operates as follows:

(1) Check whether the message m conforms to the warrant m_w. If not, stop. Otherwise, continue.

(2) Check whether Alice and Bob are specified as the original signer and the proxy signer in the warrant m_w, respectively.

(3) Recover the proxy public key $y_P = (y_A \cdot y_B)^{h(m_w, r_P)} \cdot r_P \bmod p$.

(4) Accept the proxy signature σ *if and only if* the following equality holds:

$$c \equiv h(m, m_w, \bar{r}), \quad \text{where } \bar{r} = (g^s y_P^c)^{t \cdot x_C} \bmod p. \tag{7}$$

Proxy Signature Simulation. To simulate a strong designated-verifier proxy signature σ' for any message m that conforms to the warrant m_w, Cindy picks $s' \in Z_q$ and $r' \in Z_q^*$, at random, and computes the following values:

$$r = g^{s'} y_P^{r'} \bmod p, \quad c = h(m, m_w, r),$$
$$l = r'c^{-1} \bmod q, \quad s = s'l^{-1} \bmod q, \quad t = lx_C^{-1} \bmod q. \tag{8}$$

$\sigma' = (m_w, r_P, c, s, t)$ is the simulated proxy signature for message m with respect to the proxy public key y_P. It is easy to check that σ' is also a valid proxy signature, i.e., it satisfies Eq. (7).

Now we discuss the security of the above scheme. From the results of [22], we obtain Proposition 5, and then Proposition 6 can be proved in a similar way as we did in Proposition 4.

Proposition 5. *If a valid strong designated-verifier proxy signature for Cindy can be generated without the knowledge of the proxy secret key x_P or Cindy's secret key x_C, then the computational Diffie-Hellman problem may be solved in PPT. Furthermore, the transcripts simulated by Cindy are indistinguishable from those generated by the proxy signer Bob.*

Proposition 6. *Under the CDH assumption, our strong designated-verifier proxy signature scheme is secure in the random oracle model.*

4.3 Applications

In this section, we briefly introduce two potential applications for designated-verifier proxy signatures (DVPS). Other applications are also possible. Let us first consider the following scenario. A corporate manager Alice will have vacation for one or two weeks, however, some current businesses need to be processed continuously during this period. Naturally, Alice could delegate her signing capability to several assistants to deal with each business with different customer. For example, assistant Bob, as the representative of Alice, is assigned to negotiate a business contract with customer Cindy in this period. During this procedure, some intermediate documents will be produced with digital signatures for authentication or non-repudiation. However, to protect the confidentiality and authenticity of those documents, it may be highly expected that the corresponding signatures could be validated only by the designated receiver. In this case, DVPS could be used. More specifically, Bob's proxy signatures can only be verified by Cindy. Furthermore, if non-repudiation service is required, weak DVPS could be exploited.

Another example is about on-line shopping. When a customer Cindy buys a digital product m from an Internet vendor Bob, who sells some digital products (e.g. digital music, movies, and books etc.), she needs a digital receipt from Bob to guarantee the quality, authenticity, and legality of m. This is reasonable since Cindy does not completely trust Bob and his goods.

Furthermore, Cindy would expect the receipt is bounded with not only the identity of the vendor Bob but also that of the goods producer, say Alice. With such receipts, Cindy will be convinced that digital product m is produced by Alice and sold by Bob. At the same time, to prevent Cindy from illegally re-selling m to others, Alice and Bob want the validity of Cindy's receipt can only be validated by Cindy herself. In such situations, strong designated-verifier proxy signatures, instead of ordinary digital signatures, can be used as such receipts. That is, Alice delegates her signing capability to Bob so that he can generate strong designated-verifier proxy signatures as digital receipts to all potential customers. Note that this approach cannot prevent Cindy to send a copy of digital product to her friends. To deal with this problem, one could exploit some techniques from digital right managements (DRM), such as watermarking and fingerprinting etc.

5. CONCLUSION AND FUTURE WORK

In this paper, based on the two-party Schnorr signature scheme proposed in [20], we first proposed a provably secure proxy signature scheme. Then, we extended this basic scheme into designated-verifier proxy signature (DVPS) schemes. Actually, we constructed two versions of DVPS: weak DVPS and strong DVPS. In both versions, only the designated verifier can check the validity of a proxy signature. In a weak DVPS scheme, however, the designated verifier can further convert such proxy signatures into public verifiable ones, while a strong DVPS scheme does not meet the same property even if the designated verifier's secret key is revealed willingly or unwillingly. Finally, we introduced some potential applications for DVPS.

Some other variations can be obtained directly, such as blind proxy signatures from the blind Schnorr signature [24], universally designated-verifier proxy signature scheme by using the techniques in [26], and fully distributed proxy signatures by using the techniques in [11] (though this concrete scheme is insecure [28]). Another interesting work is to design forward-secure proxy schemes.

REFERENCES

1. C. Adams and S. Farrell, Internet X.509 public key infrastructure: Certificate management protocols, RFC 2510, March 1999.
2. F. Bao, R.H. Deng, and H. Zhu, Variations of Diffie-Hellman problem, in: Information and Communicatins Security (ICICS'03), LNCS 2836, pp. 301-312, Springer-Verlag, 2003.

3. M. Bellare and P. Rogaway, Random oracles are practical: A paradigm for designing efficient protocols, in: Proc. of 1st ACM Conference on Computer and Communications Security (CCS'93), pp. 62-73, ACM Press, 1993.
4. A. Boldyreva, A. Palacio, and B. Warinschi, Secure proxy signature schemes for delegation of signing rights, Cryptology ePrint archive; http://eprint.iacr.org/2003/096, May 2003.
5. D. Boneh, The decision Diffie-Hellman problem, in: Proc. of the Third Algorithmic Number Theory Symposium (ANTS'98), LNCS 1423, pp. 48-63, Springer-Verlag, 1998.
6. D. Chaum and H. van Antwerpen, Undeniable signatures, in: CRYPTO'89, LNCS 435, pp. 212-216, Springer-Verlag, 1989.
7. D. Chaum, Zero-knowledge undeniable signatures, in: EUROCRYPT'90, LNCS 473, pp. 458-464, Springer-Verlag, 1991.
8. Y. Desmedt, C. Coutier, and S. Bengio, Special uses and abuses of the Fiat-Shamir passport protocol, in: CRYPTO'87, pp. 21-39, Springer-Verlag, 1987.
9. Y. Desmedt and M. Yung, Weakness of undeniable signature schemes, in: EUROCRYPT'91, LNCS 547, pp: 205-220, Springer-Verlag, 1991.
10. S. Goldwasser, S. Micali, and R. Rivest, A digital signature scheme secure against adaptive chosen-message attacks, SIAM Journal of Computing, 17(2): 281-308 (April 1988).
11. J. Herranz and Sáez, Verifiable secret sharing for general access structures, with applications to fully distributed proxy signatures, in: Financial Cryptography (FC'03), LNCS 2742, pp. 286-302, Springer-Verlag, 2003.
12. M. Jakobsson, Blackmailing using undeniable signatures, in: EUROCRYPT'96, LNCS 950, pp.: 425-427, Springer-Verlag, 1994.
13. M. Jakobsson, K. Sako, and R. Impagliazzo, Designated verifier proofs and their applications, in: EUROCRYPT'96, LNCS 1070, pp. 143-154, Springer-Verlag, 1996.
14. S. Kim, S. Park, and D. Won, Proxy signatures, revisited, in: Information and Communications Security (ICICS'97), LNCS 1334, pp. 223-232, Springer-Verlag, 1997.
15. B. Lee, H. Kim, and K. Kim. Secure mobile agent using strong non-designated proxy signature, in: Information Security and Privacy (ACISP'01), LNCS 2119, pp. 474-486, Springer-Verlag, 2001.
16. J.-Y. Lee, J. H. Cheon, and S. Kim, An analysis of proxy signatures: Is a secure channel necessary? in: Topics in Cryptology - CT-RSA 2003, LNCS 2612, pp. 68-79, Springer-Verlag, 2003.
17. M. Mambo, K. Usuda, and E. Okamoto, Proxy signatures: Delegation of the power to sign messages, IEICE Trans. Fundamentals, Vol. E79-A, No. 9, pp. 1338-1353 (Sep. 1996).
18. M. Mambo, K. Usuda, and E. Okamoto, Proxy signatures for delegating signing operation, in: Proc. of 3rd ACM Conference on Computer and Communications Security (CCS'96), pp. 48-57, ACM Press, 1996.
19. M. Meyers, C. Adams, D. Solo, and D. Kemp, Internet X.509 certificate request format, RFC 2511, March 1999.
20. A. Nicolosi, M. Krohn, Y. Dodis, and D. Mazieres, Proactive two-party signatures for user authentication, in: Proc. of 10th Annual Network and Distributed System Security Symposium (NDSS'03); http://www.isoc.org/isoc/conferences/ndss/.

21. D. Pointcheval and J. Stern, Security arguments for digital signatures and blind signatures, Journal of Cryptology, 13(3): 361-369 (2000).
22. S. Saeednia, S. Kremer, and O. Markowitch, An efficient strong designated verifier signature scheme, in: Information Security and Cryptology - ICISC 2003, LNCS 2971, pp. 40-54, Springer-Verlag, 2004.
23. C. Schnorr, Efficient signature generation by smart cards, Journal of Cryptography, 4(3): 161-174 (1991).
24. C. Schnorr, Security of blind discrete log signatures against interactive attacks, in: Information and Communications Security (ICICS'01), LNCS 2229, pp. 1-12, Springer-Verlag, 2001.
25. R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk, Universal designated-verifier signatures, in: ASIACRYPT'03, LNCS 2894, pp. 523-542, Springer-Verlag, 2003.
26. R. Steinfeld, H. Wang, and J. Pieprzyk, Efficient extension of standard Schnorr/RSA signatures into universal designated-verifier signatures, in: PKC 2004, LNCS 2947, pp. 86-100, Springer-Verlag, 2004.
27. G. Wang. An Attack on not-interactive designated verifier proofs for undeniable signatures, Cryptology ePrint archive; http://eprint.iacr.org/2003/243/, Nov. 2003.
28. G. Wang, F. Bao, J. Zhou, and R. H. Deng, Security analysis of some proxy signatures, in: Information Security and Cryptology - ICISC 2003, LNCS 2971, pp. 305-319, Springer-Verlag, 2004.
29. K. Zhang, Threshold proxy signature schemes, in: Information Security Workshop (ISW'97), LNCS 1396, pp. 282-290, Springer-Verlag, 1997.