Algorithms and Approaches of Proxy Signature: A Survey

Manik Lal Das[1], Ashutosh Saxena[2], and Deepak B Phatak[3]
(Corresponding author: Manik Lal Das)
Dhirubhai Ambani Institute of Information and Communication Technology Gandhinagar - 382007, India[1]
Infosys Technologies Ltd, SET Labs, Hyderabad-500019, India[2]
Computer Science & Engineering, Indian Institute of Technology, Bombay, Mumbai-400076, India[3]
(Email: maniklal_das@daiict.ac.in, Ashutosh_Saxena01@infosys.com, dbp@it.iitb.ac.in)
(Received Aug. 22, 2007; revised and accepted April 23, 2008)

Abstract

This paper reviews the recent advances on proxy signatures, discusses few notable proposals, analyzes schemes' security and efficiency, and provides an overall remark of these proposals.

Keywords: Bilinear pairings, discrete logarithm, integer factorization, proxy signature

1 Introduction

Digital signature is a cryptographic means through which the authenticity, data integrity and signer's non-repudiation can be verified. Typically, digital signature of a document is a piece of information encrypted by the signer's private key. Numerous researches have shown significant contributions to this field using various cryptographic primitives [55]. Nevertheless, there are many practical environments where digital signatures do not possess specific requirements, and thereby digital signatures appear in several other forms, namely proxy signatures, multi signatures, blind signatures, ring signatures etc. This paper aims to present a state-of-the-art discussion on recent advances of proxy signatures.

Proxy signature is a digital signature where an original signer delegates her signing capability to a proxy signer, and then the proxy signer performs message signing on behalf of the original signer. For example, a manager of a company wants to go for a long trip. She would need a proxy agent, to whom she would delegate her signing capability, and thereafter the proxy agent would sign the documents on behalf of the manager. The notion of proxy signature has been evolved over a long time, 18 years now [20]. However, the cryptographic treatment on proxy signature was introduced by Mambo et al. [53] in 1996. They first classified the proxy signature on the basis of delegation, namely *full delegation*, *partial delegation* and *delegation by warrant*, and presented a well-devised scheme. In *full delegation*, an original signer gives her private key to a proxy signer and the proxy signer signs document using original signer's private key. The drawback of proxy signature with *full delegation* is that the absence of a distinguishability between original signer and proxy signer. In *partial delegation*, the original signer derives a proxy key from her private key and hands it over to the proxy signer as a delegation capability. In this case, the proxy signer can misuse the delegation capability, because *partial delegation* cannot restrict the proxy signer's signing capability. The weaknesses of *full delegation* and *partial delegation* are eliminated by *partial delegation with warrant*. A warrant explicitly states the signers' identity, delegation period and the qualification of messages on which the proxy signer can sign, etc. Another important requirement of a proxy signature is that the revocation of delegation capability, in other words, proxy revocation. The proxy revocation is essential for the situation where original signer key is compromised or any misuse of the delegation capability is noticed. It may so happen that the original signer wants to terminate her delegation capability before its expiry. Though Mambo et al.'s [53] scheme presented an informative notion on proxy signatures and its various features; however, the scheme allows unlimited delegation, i.e., the proxy signer can sign any messages. The unlimited signing capability allows proxy signer to misuse the delegation capability.

In 1997, Kim et al. [38] proposed a scheme by restricting proxy signer signing capability using the concept of *partial delegation with warrant*. Subsequently, Zhang [85, 84] proposed threshold and non-repudiable proxy signature schemes, respectively. Ghodosi and Pieprzyk [22] analyzed the shortcomings of [85] and the same is also noticed by Lee et al. [41]. Petersen and Horster [60] proposed another notion, called self-certified keys under different trust levels and used them for creation of delegation capability, delegated signatures and proxy signatures. However, the work in [44] and [40] shows that Pertersen and Horster scheme is insecure.

In 1999, Okamoto et al. [57], for the first time, proposed proxy signature based on RSA signature scheme [63], but they considered the proxy unprotected notion (the notion is explained in Section 3). In the same year, Sun proposed two schemes [69, 73] on threshold proxy signatures. Then Lee and Kim [43] proposed a strong proxy signature scheme. Later, Viswanathan et al. [77] proposed a scheme for controlled environments.

In 2000, Sun [70, 71] proposed a multi-proxy signature scheme and a time-stamped proxy signature, respectively. Hwang et al. [34] presented a non-repudiable threshold proxy signature scheme with known signers. Subsequently, Yi et al. [83] proposed a proxy multi-signature scheme.

In 2001, Romao and da Silva [64] proposed secure mobile agent with proxy certificates. Lee et al. [44, 45] proposed two proxy signature schemes and highlighted a few applications. But, Wang et al. [80] observed security flaws of Lee et al.'s scheme [45]. Subsequently, Park and Lee [58] proposed another scheme for mobile communications.

In 2002, Shum and Wei [67] proposed a proxy signature scheme with proxy signer privacy protection, but the scheme's insecurity was found in [72].

Year 2003 saw a very impressive list of publications demonstrating a vigourous interest in the proxy signature study. In this year, a number of new schemes and improvements have been proposed [3, 9, 11, 13, 14, 28, 29, 31, 33, 35, 36, 37, 39, 40, 46, 47, 51, 65, 76, 80, 86, 87]. However, most of the schemes observed the insecurity of previously proposed schemes and proposed an improved one, which was subsequently broken by others.

In 2004, a few interesting schemes [10, 15, 30, 32, 52, 66, 74, 75, 78, 81, 82, 88] have been proposed.

In 2005, Lee and Lee [42] further addressed the security weaknesses of [67]. Wang [79] proposed a designated-verifier proxy signature scheme. Okamoto et al. [56] proposed a notion for short proxy signature using pairing. Awasthi and Lal [1] presented a multi-proxy signature scheme. Afterwards, couple of schemes [61, 89, 90] have been proposed in the same year.

In 2006, a number of schemes [2, 7, 25, 48, 50] have come into light. Lu and Huang [49] proposed a time-stamping proxy signature scheme.

In 2007 Das et al. [16] proposed a pairing-based proxy signature scheme, which does not require secure channel in key issuance stage.

In this survey, we review several notable proxy signature schemes categorizing them into different constructions based on their security assumptions. The organization of this survey is as follows. In the next Section, we give a mathematical background for general readership. Section 3 discusses the security properties of proxy signature. Section 4 details the constructions of various proxy signatures. Section 5 reviews some of the notable proxy signature schemes. Section 6 gives concluding remarks of our observations.

2 Preliminaries

2.1 Discrete Logarithm Problem and Its Applications

The discrete logarithm is the inverse of discrete exponentiation in a finite cyclic group. Suppose G be a finite multiplicative cyclic group of order a large prime q. Let g be a generator of G. Then every element y of G can be written in the form $y = g^k$ for some integer k. The discrete logarithm of y is k and is written as $\log_g y = k \bmod q$.

In 1976, Diffie and Hellman [17] revolutionized the cryptography and proposed a key exchange protocol without using secure channel, where the security of the protocol relies on discrete logarithm problem (DLP). Since then, several key exchange protocols [6], public key encryption [8, 55], and signature schemes [18, 23, 68] have been proposed in which the security assumptions trust on the hardness of solving the DLP. Below we present a signature scheme, named Schnorr's signature, which is being taking into account while constructing several proxy signature schmes discussed in Section 5.1.

The Schnorr signature scheme: The scheme S_{sch} is based on DLP and works as follows:

- `Setup` ($\mathcal{SP}_{dlp}$): Inputs 1^k; and outputs **params-dlp**. The **params-dlp** consists of primes q and l such that $2^{k-1} \leq q < 2^k$, an element $g \in \mathbb{Z}_q^*$ of order l that divides $q-1$, and a hash function $h : \{0,1\}^* \rightarrow \mathbb{Z}_l$.

- `KeyGen` ($\mathcal{KG}_{dlp}$): The users agree on a group G (multiplicative group of integers modulo q) for some prime q with generator g of prime order l in which the DLP is hard. The user chooses a private key $x \in \mathbb{Z}_l$. The public key is generated as $y = g^x \bmod q$.

  In other words, user public key $\leftarrow \mathcal{KG}_{dlp}$(**params-dlp**, user private key), i.e.

  $$y \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x).$$

- `Sign` ($\mathcal{S}_{dlp}$): To sign a message m, the signer has to choose a random $t \in \mathbb{Z}_l$ and compute $r = g^t \bmod q$. Then compute $c = h(m, r)$ and $\sigma = (t - xc) \bmod l$. The signature on message m is (σ, c). In other words, $\sigma \leftarrow \mathcal{S}_{dlp}$(**params-dlp**, (t, r), x, m).

- `Verify` ($\mathcal{V}_{dlp}$): The verifier computes $r' = g^\sigma y^c \pmod q$ and $c' = h(m, r')$. If $c' = c$ then the signature is valid, else the signature is invalid. In other words,

  $$\textbf{Result} \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y, \sigma, m),$$

  where **Result** $\in \{Valid, Invalid\}$.

The Schnorr's signature scheme is proven secure [68] under the assumption that DLP is hard.

2.2 Bilinear Pairings

Bilinear pairings were first introduced to elliptic curve cryptography for destructive methods like the MOV reduction [54]. With the help of Weil pairing, the authors of [54] showed a way to reduce the DLP on supersingular elliptic curves to the DLP of an extension of the underlying finite field. Later, Frey and Ruck [19] extended the attack to general elliptic curves with the Tate pairing. However, the Weil pairing and the Tate pairing can also be used as a constructive tool for cryptography [4, 5, 12, 27].

Suppose G_1 is an additive cyclic group generated by P of prime order q, and G_2 is a multiplicative cyclic group of the same order. A map $\hat{e} : G_1 \times G_1 \rightarrow G_2$ is called a bilinear mapping if it satisfies the following properties:

- Bilinear: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in \mathbb{Z}_q^*$;
- Non-degenerate: There exist $P, Q \in G_1$ such that $\hat{e}(P, Q) \neq 1$;
- Computable: There exist efficient algorithm to compute $\hat{e}(P, Q)$ for all $P, Q \in G_1$. In general, G_1 is a group of points on an elliptic curve and G_2 is a multiplicative subgroup of a finite field.

Definition 1. *Discrete Logarithm Problem (DLP): Given $Q, R \in G_1$, find an integer $x \in \mathbb{Z}_q^*$ such that $R = xQ$.*

Definition 2. *Decisional Diffie-Hellman Problem (DDHP): Given (P, aP, bP, cP) for $a, b, c \in \mathbb{Z}_q^*$, determine whether $c \equiv ab \bmod q$.*

The advantage of any probabilistic polynomial time (PPT) algorithm $\mathcal{A}$ in solving DDHP in G_1 is defined as

$$\begin{aligned} \mathtt{Adv}_{\mathcal{A},G_1}^{DDH} &= [\mathtt{Prob}[\mathcal{A}(P, aP, bP, cP) = 1] \\ &\quad -\mathtt{Prob}[\mathcal{A}(P, aP, bP, abP) = 1] : a, b \in \mathbb{Z}_q^*]. \end{aligned}$$

The DDHP is easy in G_1, since it is easy to compute $\hat{e}(aP, bP) = \hat{e}(P, P)^{ab}$ and decide whether $\hat{e}(P, P)^{ab} = \hat{e}(P, P)^{c}$.

Definition 3. *Computational Diffie-Hellman Problem (CDHP): Given (P, aP, bP) for $a, b \in \mathbb{Z}_q^*$, compute abP.*

The advantage of any probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$ in solving CDHP in G_1, is defined as $\mathtt{Adv}_{\mathcal{A},G_1}^{CDH} = \mathtt{Prob}[\mathcal{A}(P, aP, bP) = abP : a, b \in \mathbb{Z}_q^*] \geq \epsilon$. For every PPT algorithm $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A},G_1}^{CDH}$ is negligible ϵ. There is no known efficient algorithm which can solve CDHP in G_1.

Definition 4. *Gap Diffie-Hellman (GDH) group: A prime order group G_1 is a GDH group if there exists an efficient polynomial-time algorithm which solves the DDHP in G_1 and there is no PPT algorithm which solves the CDHP with non-negligible probability of success.*

The domains of bilinear pairings provide examples of GDH groups. The MOV reduction [54] provides a method to solve DDHP in G_1, but there is no known efficient algorithm for CDHP in G_1.

Definition 5. *Bilinear Diffie-Hellman Problem (BDHP): Given (P, aP, bP, cP) for $a, b, c \in \mathbb{Z}_q^*$, compute $\hat{e}(P, P)^{abc}$.*

Definition 6. *Weak Diffie-Hellman Problem (WDHP): Given (P, Q, aP) for $a \in \mathbb{Z}_q^*$, compute aQ.*

Since most of our discussions on pairing-based proxy signatures are surrounded by Hess's signature scheme [27], we briefly discuss the scheme as follows.

The Hess signature scheme: The scheme $\mathcal{S}_{hess}$ is based on CDHP and works as follows:

- `Setup`($\mathcal{SP}_{cdhp}$): It takes 1^k and master-key s as input; and outputs **params-cdhp**. The **params-cdhp** includes groups G_1, G_2 of order prime q; a generator $P \in G_1$; a bilinear map $\hat{e} : G_1 \times G_1 \rightarrow G_2$; map-to-point $H : \{0,1\}^* \rightarrow G_1$, hash function $h : \{0,1\}^* \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ and public key of a trusted party, say Key Generation Center (KGC) ($Pub_{KGC} = sP$). The KGC keeps s secret.
- `KeyGen` ($\mathcal{KG}_{cdhp}$): It takes **params-cdhp**, user (with identity ID) public key $Pub_{ID} = H(ID)$ as input; outputs user private key $S_{ID} = sPub_{ID}$, i.e., $S_{ID} \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, Pub_{ID}).
- `Sign` ($\mathcal{S}_{cdhp}$): To sign a message m, the signer chooses an arbitrary $P_1 \in G_1$, picks a random $t \in \mathbb{Z}_q^*$ and computes

$$\begin{aligned} r &= \hat{e}(P_1, P)^t, \\ c &= h(m, r), \\ \sigma &= cS_{ID} + tP_1. \end{aligned}$$

  The signature on message m is the tuple (c, σ). In other words, $\sigma \leftarrow \mathcal{S}_{cdhp}$(**params-cdhp**, (t, r, c), S_{ID}, m).
- `Verify` ($\mathcal{V}_{cdhp}$): The signature (c, σ) is verified by the following checking: Compute $r' = \hat{e}(\sigma, P) \cdot \hat{e}(H(ID), -Pub_{KGC})^c$. Accept the signature if $c = h(m, r')$. Else reject the signature. In other words, **Result**$\leftarrow \mathcal{V}_{cdhp}$(**params-cdhp**, Pub_{ID}, Pub_{KGC}, σ, (c, m)), where **Result** $\in \{Valid, Invalid\}$.

The Hess's signature scheme is proven secure against existential forgery on adaptive chosen-message attack under the assumption that CDHP is hard.

2.3 Integer Factorization Problem

The integer factorization (also known as prime decomposition) problem (IFP) is: *Input* a large positive integer; *output* it as a product of prime numbers. The problem

holds a strong security assumption in cryptography, complexity theory, and quantum computers. Based on IFP, the most widely used scheme for public key encryption and signature is RSA [63]. There are several algorithms, protocols, and products where security relies on IFP.

The RSA signature scheme: The signature scheme $\mathcal{S}_{rsa}$ works as follows:

- `Setup`($\mathcal{SP}_{rsa}$): Inputs 1^k, secret large primes p, q; and outputs **params-rsa**. The **params-rsa** is public and consists of a modulus $N = pq$ and hash function $h : \{0,1\}^* \rightarrow \mathbb{Z}_N$.
- `KeyGen` ($\mathcal{KG}_{rsa}$): Choose public key e such that $1 < e < \phi(N)$ which is co-prime to $\phi(N)$, where $\phi(N) = (p-1)(q-1)$. Compute private key d such that $de \equiv 1 \bmod \phi(N)$. Procedurally, $d \leftarrow \mathcal{KG}_{rsa}$(**params-rsa**, e).
- `Sign` ($\mathcal{S}_{rsa}$): To sign a message m, compute $\sigma = h(m)^d \bmod N$. The signature of a message m is the tuple (m, σ). In other words, $\sigma \leftarrow \mathcal{S}_{rsa}$(**params-rsa**, d, m).
- `Verify` ($\mathcal{V}_{rsa}$): Compute $m' = \sigma^e \bmod N$. The signature is valid if $m' = h(m)$.

In other words, **Result**$\leftarrow \mathcal{V}_{rsa}$(**params-rsa**, e, σ, m), **Result** $\in \{Valid, Invalid\}$.

3 Security Properties of Proxy Signature

Desirable security properties of proxy signatures have been evolved from the introduction of proxy signature. A widely accepted list of required security properties is given below:

- Strong unforgeability: A designated proxy signer can create a valid proxy signature on behalf of the original signer. But the original signer and other third parties cannot create a valid proxy signature.
- Strong identifiability: Anyone can determine the identity of corresponding proxy signer from the proxy signature.
- Strong undeniability: Once a proxy signer creates a valid proxy signature on behalf of the original signer, he cannot deny the signature creation.
- Verifiability: The verifier can be convinced of the signers' agreement from the proxy signature.
- Distinguishability: Proxy signatures are distinguishable from the normal signatures by everyone.
- Secrecy: The original signer's private key cannot be derived from any information, such as the shares of the proxy key, proxy signatures, etc.
- Prevention of misuse: The proxy signer cannot use the proxy key for other purposes than it is made for. That is, he cannot sign message with the proxy key that have not been defined in the warrant. If he does so, he will be identified explicitly from the warrant.

3.1 Classification of Proxy Signature

According to the nature of delegation capability, proxy signature can be classified as proxy-unprotected, proxy-protected and threshold notions. This differentiation is important in practical applications, since it enables proxy signature schemes to avoid potential disputes between the original signer and the proxy signer.

3.1.1 Proxy-Unprotected Notion

The scenario exists when an original signer gives her signing rights (full delegation with warrant) to a proxy signer. The original signer sends a signed warrant to the proxy signer, who then uses this information to generate proxy signatures by executing a standard signature scheme. When a proxy signature is sent, the recipient checks its validity according to the corresponding standard signature verification process. As the proxy signer does not append his private key on top of the received delegation, a dishonest original signer can sign the message and later claim that the signature was created by the proxy signer. This type of proxy signature primarily lacks strong unforgeability property.

3.1.2 Threshold Notion

In a threshold proxy signature, the proxy key is shared by a group of n proxy signers. In order to produce a valid proxy signature on a given message m, individual proxy signer produces his partial signature on that message, and then combines them into a full proxy signature on message m.

In a (t, n) threshold proxy signature scheme, the original signer delegates her signing capability to a proxy group of n members. Any t or more proxy signers of the group can cooperatively issue a proxy signature on behalf of the original signer, but $(t-1)$ or less proxy signers cannot forge a signature.

4 Models of Proxy Signature

Based on the security assumptions on various proxy signature schemes, we categorize the existing schemes into four different constructions: DLP-based proxy signature, RSA-based proxy signature, ECDSA-based proxy signature, and Pairing-based proxy signature. The schemes consist of delegation capability generation, delegation capability verification, proxy key generation, proxy signature generation and proxy signature verification.

4.1 DLP-based Proxy Signature

The participants involved in the model are:

- An original signer, who delegates her signing capability to a proxy signer.
- A proxy signer, who signs the message on behalf of the original signer.
- A verifier, who verifies the proxy signature and decides to accept or reject.
- A trusted party, who certifies the public key.

DLP-based Proxy Signature Model: An original signer selects a private key x_o and computes her public key y_o as

$$y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o).$$

A proxy signer selects a private key x_p and computes his public key y_p as

$$y_p \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_p).$$

- Delegation capability generation: It takes **params-dlp**, original signer chosen parameters (k_o, r_o), original signer private key x_o, a warrant ω as input; and outputs signature σ_o on ω. Procedurally,

$$\sigma_o \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_o, r_o), x_o, \omega).$$

- Delegation capability verification: It takes **params-dlp**, y_o, ω, σ_o as input; and outputs **Result**, where **Result**$\in \{Valid, Invalid\}$, i.e., **Result**$\leftarrow \mathcal{V}_{dlp}$(**params-dlp**, y_o, σ_o, ω).
- Proxy key generation($\texttt{PKeyGen}_{dlp}$): It takes **params-dlp**, σ_o, x_p and random number as input; and outputs proxy key ρ_p. Typically, the proxy signer uses simple arithmetic operation to form a proxy key $\rho_p = y_o\sigma_o + x_p y_p \bmod q$. Procedurally, $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}$(**params-dlp**, σ_o, x_p, *pub-params*[1]).
- Proxy signature generation: It takes **params-dlp**, proxy key ρ_p and message m as input; outputs signature σ_p on m, i.e, $\sigma_p \leftarrow \mathcal{S}_{dlp}$(**params-dlp**, ρ_p, m).
- Proxy signature verification: It takes **params-dlp**, y_o, y_p, m and σ_p as input; outputs **Result**, i.e., **Result**$\leftarrow \mathcal{V}_{dlp}$(**params-dlp**, (y_o, y_p), σ_p, m).

4.2 RSA-based Proxy Signature

The participants involved in the model are:

- An original signer, who delegates her signing capability to a proxy signer.
- A proxy signer, who signs the message on behalf of the original signer.
- A verifier, who verifies the proxy signature and decides to accept or reject.
- A trusted party, who certifies the public key.

RSA-based Proxy Signature Model: An original signer selects a public key y_o and computes her private key x_o as

$$x_o \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_o).$$

A proxy signer selects a public key y_p and computes his private key x_p as

$$x_p \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_p).$$

- Delegation capability generation: It takes **params-rsa**, x_o, a warrant ω as input; and outputs signature σ_o on ω, Procedurally, $\sigma_o \leftarrow \mathcal{S}_{rsa}$(**params-rsa**, x_o, ω).
- Delegation capability verification: It takes **params-rsa**, y_o, ω, σ_o as input; and outputs **Result**. That is, **Result**$\leftarrow \mathcal{V}_{rsa}$(**params-rsa**, y_o, σ_o, ω), where **Result** $\in \{Valid, Invalid\}$.
- Proxy signature generation: It takes **params-rsa**, σ_o, x_p and message m as input; outputs signature σ_p on m, i.e, $\sigma_p \leftarrow \mathcal{S}_{rsa}$(**params-rsa**, x_p, (σ_o, m)) Proxy signature verification: It takes **params-rsa**, y_o, y_p, m and σ_p as input; and outputs **Result**, i.e., **Result**$\leftarrow \mathcal{V}_{rsa}$(**params-rsa**, (y_o, y_p), σ_p, m).

4.3 Pairing-based Proxy Signature

The participants involved in the model are:

- An original signer, who delegates her signing capability to a proxy signer.
- A proxy signer, who signs the message on behalf of the original signer.
- A verifier, who verifies the proxy signature and decides to accept or reject.
- A trusted party, who issues user private key.

Pairing-based Proxy Signature Model: The original signer generates her public key $y_o = H(ID_o)$, and computes private key $x_o \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_o), where ID_o is original signer's identity.

The proxy signer generates his public key $y_p = H(ID_p)$, and computes private key

$$x_p \leftarrow \mathcal{KG}_{cdhp}(\textbf{params-cdhp}, y_p),$$

where ID_p is proxy signer's identity.

[1] *pub-params* include signers' public keys, random numbers, warrant, etc.

- Delegation capability generation: Takes **params-cdhp**, x_o and a warrant ω as input; outputs signature σ_o on ω, i.e., $\sigma_o \leftarrow \mathcal{S}_{cdhp}$(**params-cdhp**, (k_o, r_o, c_o), x_o, ω).
- Delegation capability verification: It takes **params-cdhp**, y_o, ω and σ_o as input; and outputs **Result**, i.e., **Result**$\leftarrow \mathcal{V}_{cdhp}$(**params-cdhp**, (y_o, Pub_{KGC}), σ_o, (c_o, ω)), where **Result** $\in \{Valid, Invalid\}$.
- Proxy key generation: It takes **params-cdhp**, σ_o, x_p and random number as input; and outputs proxy key $\rho_p \leftarrow$ $\texttt{PKeyGen}_{cdhp}$(**params-cdhp**, σ_o, (*user-params*[2]), x_p).
- Proxy signature generation: Takes **params-cdhp**, proxy key ρ_p and message m as input; outputs signature σ_p on m, i.e., $\sigma_p \leftarrow \mathcal{S}_{cdhp}$(**params-cdhp**, (k_p, r_p), ρ_p, m).
- Proxy signature verification: It takes **params-cdhp**, y_o, y_p, m and σ_p as input; outputs **Result**$\leftarrow \mathcal{V}_{cdhp}$(**params-cdhp**, (y_o, y_p, Pub_{KGC}), σ_p, (c_p, m, ω)).

5 Review of Some Notable Proxy Signature Schemes

5.1 DLP-based Proxy Signature Schemes

5.1.1 Mambo, Usuda and Okamoto (1996)

First classified the proxy signature on the basis of the degree of delegation, and proposed a well-devised scheme. In their scheme both proxy unprotected and proxy protected notions are envisaged. As we are more focused on proxy-protected scheme, here we discuss the proxy-protected notion.

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}$(**params-dlp**, x_o).
- Bob picks a private key x_p and generates public key $y_p \leftarrow \mathcal{KG}_{dlp}$(**params-dlp**, x_p).
- Delegation capability generation: Alice chooses a random number $k_o \in \mathbb{Z}^*_{q-1}$ and computes $r_o = g^{k_o} \bmod q$. Alice computes $\sigma_o \leftarrow \mathcal{S}_{dlp}$(**params-dlp**, (k_o, r_o), x_o).
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_o, \sigma_o).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}$(**params-dlp**, σ_o, x_p, *pub-params*).
- Proxy signature generation: The proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_p, r_p), \rho_p, m).$$

- Proxy signature verification: The verifier accepts the proxy signature if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (y_o, y_p), \sigma_p, m).$$

- Security: The underlying security of the scheme is based on the hardness of DLP. But, the scheme has two weaknesses. Firstly, unlimited delegation, i.e., Bob can sign any messages on behalf of Alice, because Alice has delegated unlimited signing rights to Bob. Secondly, proxy transfer, i.e., Bob transfers Alice's delegation power to any other party who can sign any message on behalf of Alice. In other words, the scheme does not satisfy the prevention of misuse security property.

5.1.2 Kim, Park and Won (1997)

Proposed a proxy signature scheme using partial delegation with warrant and a proxy signature scheme using threshold delegation.

Assumption: DLP is hard.

Scheme for Partial Delegation with Warrant (PDW):

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}$(**params-dlp**, x_o).
- Bob picks a private key x_p and generates public key $y_p \leftarrow \mathcal{KG}_{dlp}$(**params-dlp**, x_p).
- Delegation capability generation: Alice chooses a random number $k_o \in \mathbb{Z}^*_{q-1}$ and computes $r_o = g^{k_o} \bmod q$.

  Then, Alice computes $\sigma_o \leftarrow \mathcal{S}_{dlp}$(**params-dlp**, (k_o, r_o), x_o, ω).
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_o, \sigma_o, \omega).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}$(**params-dlp**, σ_o, x_p, *pub-params*).
- Proxy signature generation: The proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_p, r_p), \rho_p, m).$$

- Proxy signature verification: The verifier accepts the proxy signature if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (y_o, y_p), \sigma_p, (\omega, m)).$$

[2]The *user-params* includes signers' public keys, random numbers, warrant, etc.

Table 1: Conventions and notation for DLP-based proxy signature schemes

Alice	Original signer
Bob	Proxy signer
q	A large prime
$\mathbb{Z}_q$	Set of integers modulo q
$\mathbb{Z}_q^*$	Multiplicative group of $\mathbb{Z}_q$
g	Generator of large order in $\mathbb{Z}_q^*$
x_o, x_p	Private key of Alice and Bob, respectively
y_o	Public key of Alice, $y_o = g^{x_o} \bmod q$
y_p	Public key of Bob, $y_p = g^{x_p} \bmod q$
ω	A warrant
$h(.)$	A collision-resistant one-way hash function

Scheme for Threshold Delegation: In the threshold delegation, Alice sends her delegation to a proxy group so that the proxy signer's power is shared to sign a message.

- Alice picks private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Each proxy signer acts as a dealer with a random secret u, chooses a random polynomial such that $f(x) = u + a_1 x + \cdots + a_{t-1}x^{t-1} \bmod (q-1)$. The proxy group keys are generated as follows:
  - Public keys: $g^u \bmod q$, $g^{a_1} \bmod q$, $\cdots$, $g^{a_{t-1}} \bmod q$.
  - Private keys: $x_{p,i} = u + a_1 i + \cdots + a_{t-1} i^{t-1}$; $i = 1, 2, \cdots, t$.
- Delegation capability generation: Alice chooses a random number $k_o \in \mathbb{Z}_{q-1}^*$ and computes $r_o = g^{k_o} \bmod q$. Then Alice computes $\sigma_o \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_o, r_o), x_o, \omega)$.
- Proxy sharing: To share σ_o in a threshold manner with threshold t, Alice chooses random $b_j \in \mathbb{Z}_{q-1}; j = 1, 2, \cdots, t-1$, and publishes the values $B_j = g^{b_j}, j = 1, 2, \cdots, t-1$. Then, she computes the proxy share σ_i as
$$\sigma_i = f'(i) = \sigma_o + b_1 i + \cdots + b_{t-1} i^{t-1}.$$
- Delegation capability verification: Each proxy signer accepts σ_i if and only if
$$g^{\sigma_i} = y_o^{h(\omega, r_o)} r_o \cdot \prod_{j=1}^{t-1} B_j^{(i^j)} \bmod q.$$
- Proxy key generation: Each proxy signer computes proxy key as
$$\rho_{p,i} \leftarrow \texttt{PKeyGen}_{dlp}(\textbf{params-dlp}, \sigma_i, x_{p,i}, pub-params).$$
- Proxy signature generation: To sign a message m, each proxy signer computes $v = h(l, m)$, where $l = g^r \bmod q$ (r is secret to the proxy signer). Then, the proxy signer computes $\lambda_i = s_i + \sigma_i v \bmod (q-1)$ and reveals λ_i, where $s_i = f(i) = r + a_1 i + \cdots + a_{t-1} i^{i-1}$.

  On validating λ_i, each proxy signer computes σ satisfying $\sigma = r + \sigma_o v = f(0) + f'(0)v \bmod (q-1)$ by applying Lagrange formula to λ_i. The proxy signature on m is the tuple $(\omega, r_o, m, \sigma, v)$.
- Proxy signature verification: The verifier checks whether $v' = g^{\sigma} \cdot (y_o \cdot y_p)^{h(\omega, r_o)} r_o)^{-v} \bmod q$, and then whether $v = h(v', m)$.
- Security: To the best of our knowledge the proposed PDW scheme is still unbroken. However, the intuition in this scheme that using warrant does not require proxy revocation is not correct. There are many situations where proxy revocation is a must though warrant explicitly states the validity and restricts the message signing. Sun et al. [73] showed that the above threshold delegation approach is insecure.

5.1.3 Zhang (1997a)

Proposed a threshold and non-repudiable proxy signature, where both the original signer and proxy signer cannot falsely deny their signature.

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Assume that there is a group of n proxy signers p_i, $i = 1, \cdots, n$.
- Proxy key generation: Alice picks $k_o \in \mathbb{Z}_{q-1}$, computes $R = g^{k_o} \bmod q$, and broadcasts R. The proxy signer randomly selects $\alpha_i \in \mathbb{Z}_{q-1}$, computes $y_{p_i} = g^{\alpha_i} R \bmod q$, checks whether $y_{p_i} \in \mathbb{Z}_{q-1}^*$ and if it holds then broadcasts y_{p_i}.
- Alice computes $\hat{R} = \prod_{i=1}^{n} y_{p_i}$, and $\hat{s} = n^{-1}\hat{R}x_o + k \bmod (q-1)$ and broadcasts $\hat{s}$. Then, each proxy signer computes $\hat{R} = \prod_{i=1}^{n} y_{p_i}, \sigma_{p_i} = \hat{s} + \alpha_i \bmod (q-1)$, and checks if the equality holds: $g^{\hat{s}} = y^{n^{-1}\hat{R}} R \bmod q$. If it holds, the proxy signer accepts σ_{p_i} as a valid proxy share.
- The threshold proxy signature and verification are done in similar approaches as in the schemes [21, 26].

- Security: The articles [41, 73] pointed out some weaknesses in Zhang's threshold proxy signatures [85]. Later, some additional attacks are commented in [22].

5.1.4 Petersen and Horster (1997)

Proposed a scheme for self-certified keys issuance under different trust level and used them for delegation of signing rights and delegated signatures, proxy signatures.

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Bob picks a private key x_p and generates public key $y_p \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_p)$.
- Delegation capability generation: Alice picks a random number $k_o \in \mathbb{Z}^*_{q-1}$, computes $r_o = g^{k_o} \bmod q$. Then, Alice computes $\sigma_o \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_o, r_o), x_o, ProxyID)$.
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_o, \sigma_o, ProxyID).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}(\textbf{params-dlp}, \sigma_o, x_p, pub\text{-}params)$.
- Proxy signature generation: The proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, \rho_p, (m, ProxyID)).$$

- Proxy signature verification: The verifier accepts the proxy signature if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (y_o, y_p), \sigma_p, (m, ProxyID)).$$

- Security: The scheme has three weaknesses. Firstly, the proxy signer can deny his signature creation later because a proxy signature does not contain any authentic information of proxy signer. Secondly, the proxy signer gets a proxy key pair (x_p, y_p) from the original signer. He can deny his signature by showing that the proxy signature is created by the original signer with the name of him. Thirdly, the original signer sends the signing rights to a proxy signer without any agreement or warrant. The original signer can argue that the proxy signature is not valid for the concerned message. Moreover, the schemes in [40, 44] further pointed out some more weaknesses of this scheme.

5.1.5 Sun, Lee and Hwang (1999)

Proposed a (t, n) threshold proxy signature based on Zhang's threshold scheme [85].

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Assume that there is a group of n proxy signers p_i, $i = 1, \cdots, n$. Each proxy signer p_i has a private key $x_{p_i} \in \mathbb{Z}_{q-1}$ and a public key $y_{p_i} \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_{p_i})$.
- Proxy generation: Alice picks $k_o \in \mathbb{Z}_{q-1}$, computes $R = g^{k_o} \bmod q$, and broadcasts R. Then each proxy signer randomly selects $\alpha_i \in \mathbb{Z}_{q-1}$, computes $r_{p_i} = g^{\alpha_i} R \bmod q$.

  Alice computes $\hat{R} = \prod_{i=1}^{n} r_{p_i} \bmod q$, and $\hat{s} = n^{-1} x_o h(\hat{R}, PGID) + k_o \bmod (q-1)$ and broadcasts $\hat{s}$, where $PGID$ is the proxy group identity that records the proxy status, the event mark of the proxy share generation, the expiration time of the delegation, the identities of original signer and proxy signers. After validating $\hat{s}$, each proxy signer performs a (t, n) verifiable threshold secret sharing scheme [59], and acts as a dealer to distribute proxy subshares to other $n-1$ proxy signers for generating their valid proxy shares. Each proxy signer p_i selects a $(t-1)$-degree polynomial $f_i(x) = s_i + a_{i,1}x + a_{i,2}x^2 + \cdots + a_{i,t-1}x^{t-1} \bmod (q-1)$, where $s_i = \hat{s} + \alpha_i + x_{p_i} h(\hat{R}, PGID) \bmod (q-1)$. Then p_i sends the proxy sub-share $f_i(j)$ to proxy signer p_j(for $1 \le j \le n$ and $j \ne i$) vis a secure channel. In addition, p_i also broadcasts $g^{a_{i,1}}, \cdots, g^{a_{i,t-1}}$.

  After validating all $f_j(i)$, p_i computes $x'_i = \sum_{j=1}^{n} f_j(i) \bmod (q-1)$ as his proxy share. Let $f(x) = \sum_{j=1}^{n} f_j(x) \bmod (q-1)$. This proxy share can be written as $x'_i = f(i)$ and will be used for generating proxy signatures. The shared secret key is regarded as $f(0) = \hat{n}s + \sum_{i=1}^{n} \alpha_i + \sum_{i=1}^{n} x_{p_i} h(\hat{R}, PGID) = \sum_{i=1}^{n} (\alpha_i + k_o) + \sum_{i=0}^{n} x_{p_i} h(\hat{R}, PGID) \bmod (q-1)$.
- Proxy signature generation: Each participant proxy signer p_i $(1 \le i \le t)$ performs a (t, t) verifiable secret sharing scheme by randomly choosing a $(t-1)$-degree polynomial $f'_i(x) = \sum_{j=0}^{t-1} a'_{i,j} x^j \bmod (q-1)$ and broadcasts $c'_{i,j} = g^{a'_{i,j}} \bmod q$ for $j = 0, 1 \cdots, t-1$. Then p_i computes $f'_i(j)$ and sends it to p_j via a secure channel for $1 \le j \le n$ and $j \ne i$. After

validating each $f_i'(j)$, each participant proxy signer p_i computes $x_i'' = f(i) = \sum_{j=1}^{t} f_j'(i) \bmod (q-1)$, where $f'(x) = \sum_{j=1}^{t} f_j'(x) \bmod (q-1)$ and $Y = \prod_{k=1}^{t} c'_{k,0} \bmod q$. Finally, each p_i computes and broadcasts $T_i = x_i' h(m) + x_i'' Y \bmod (q-1)$.

On validating T_i, each p_i computes $T = f(0)h(m) + f'(0)Y \bmod (q-1)$ from T_j by applying Lagrange's interpolating polynomial, where m is the message. The proxy signature on message m is the tuple $(\hat{R}, PGID, Y, T)$.

- Proxy signature verification: The verifier accepts the proxy signature if and only if

$$g^T = \left(\left(y_o \prod_{i=1}^{n} y_i \right)^{h(\hat{R}, PGID)} \hat{R} \right)^{h(m)} Y^Y \bmod q.$$

- Security: Hsu et al. [31] and Shao [66] noticed that Sun et al.'s scheme is not secure, it lacks coalition attack.

5.1.6 Lee, Kim and Kim (2001b)

Proposed a scheme in which a mobile agent is constructed using non-designated proxy signature which represents both the original signer's (customer) and the proxy signer's (remote server) signatures. The work provides Schnorr-based and RSA-based constructions for secure mobile agent. Here, we give the Schnorr-based construction, and the RSA-based construction is given in the next section.

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Bob picks a private key x_p and generates public key $y_p \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_p)$.
- Delegation capability generation: Alice chooses a random number $k_o \in \mathbb{Z}_{q-1}^*$ and computes $r_o = g^{k_o} \bmod q$. Then she computes $\sigma_o \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_o, r_o), x_o, \omega)$.
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_o, \sigma_o, \omega).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}(\textbf{params-dlp}, \sigma_o, x_p, \textit{public-parameters})$.
- Proxy signature generation: The proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_p, r_p), \rho_p, m).$$

- Proxy signature verification: The verifier accepts the proxy signature if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (y_o, y_p), \sigma_p, (m, \omega)).$$

- Security: Wang et al. [80] showed that the scheme is insecure against transferring and forgery attacks.

5.1.7 Boldyreva, Palacio and Warinschi (2003)

Proposed a formal security notion for proxy signature. At the same time, they proposed a provable secure scheme, called *triple* Schnorr proxy signature scheme, which is an enhanced version of the PDW scheme [38].

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Bob picks a private key x_p and generates public key $y_p \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_p)$.
- Delegation capability generation: Alice chooses a random number $k_o \in \mathbb{Z}_{q-1}^*$ and computes $r_o = g^{k_o} \bmod q$. Then computes $\sigma_o \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_o, r_o), x_o, (\omega, y_o, y_p))$.
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_o, \sigma_o, (\omega, y_o, y_p)).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}(\textbf{params-dlp}, \sigma_o, x_p, \textit{pub-params})$.
- Proxy signature generation: The proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_p, r_p, y_o, y_p), \rho_p, m)).$$

- Proxy signature verification: The verifier accepts the proxy signature if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (y_o, y_p), \sigma_p, m)).$$

- Security: The scheme is based on Kim et al.'s PDW scheme [38]. The scheme did not consider warrant in the proxy signing phase which leads unlimited delegation, that is, the scheme suffers from delegation misuse.

5.1.8 Li, Tzeng and Hwang (2003)

Proposed a generalized version $(t_1/n_1 - t_2/n_2)$ proxy signature scheme. The $(t_1/n_1 - t_2/n_2)$ proxy signature scheme allows t_1 out of n_1 original signers to delegate their signing capability to a designated proxy group of t_2 out of n_2 proxy signers. The proxy group of proxy signers can cooperatively generate the proxy signature on behalf of the original group. Any verifier can verify the proxy signature on the message with the knowledge of the identities of the actual original signers and the actual proxy signers.

Assumption: DLP is hard.

- Let the scheme consists of n_1 original signers and n_2 proxy signers.
- For $i = 1, 2, \cdots, n_1$, the original signer chooses a private key x_{o_i} and generates public key $y_{o_i} \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_{o_i})$.
- For $j = 1, 2, \cdots, n_2$, the proxy signer chooses a private key x_{p_j} and generates public key $y_{p_j} \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_{p_j})$.
- Delegation capability generation: For $i = 1, 2, \cdots, t_1$ $(< n_1)$, the original signer chooses a random number $k_{o_i} \in \mathbb{Z}^*_{q-1}$ and computes $r_{o_i} = g^{k_{o_i}} \bmod q$. Then the original signer computes $\sigma_{o_i} \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_{o_i}, r_{o_i}), x_{o_i}, \omega)$. A designated clerk (any one of the original signers) verifies the individual proxy shares as whether

  $$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_{o_i}, \sigma_{o_i}, \omega).$$

  If it holds, the clerk combines the individual proxy shares as $\sigma_o = \sum_{i=1}^{t_1} \sigma_{o_i} \bmod q - 1$. The final proxy share is the tuple (ω, K, σ_o), where $K = \prod_{i=1}^{t_1} r_{o_i}$.
  Delegation capability verification: Bob accepts σ_o if and only if

  $$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (K, y_o), \sigma_o, \omega).$$

- Proxy signature generation: Each proxy signer selects a random integer $k_{p_j} \in \mathbb{Z}^*_{q-1}$, computes $r_{p_j} = g^{k_{p_j}} \bmod q$, and broadcast r_{p_j}. Then, each proxy signer computes $R = \prod_{j=1}^{t_2} r_{p_j} \bmod q$ and $\sigma_{p_j} = k_{p_j} R + (\sigma_o t_2^{-1} + x_{o_i} y_{o_i})^{h(m,R,ProxyID)} \bmod (q-1)$, where t_2 is the threshold value of the proxy signers group. A designated clerk verifies the individual proxy signature as whether

  $$g^{\sigma_{p_j}} = r_{p_j}^R ((K^K \prod_{i=1}^{t_1} y_{o_i}^{y_{o_i} h(\omega,K)})^{t_2^{-1}} y_{p_j}^{y_{p_j}})^{h(m,R,ProxyID)} \bmod q.$$

  If it does, the clerk combines the individual proxy signature of m as $\sigma = \sum_{j=1}^{t_2} \sigma_{p_j} \bmod (q-1)$. The proxy signature of m is $(\omega, K, m, R, \sigma)$.
- Proxy signature verification: A verifier checks the validity of the proxy signature on m whether

  $$g^{\sigma} = R^R (K^K \prod_{i=1}^{t_1} y_{o_i}^{y_{o_i} h(\omega,K)} \prod_{j=1}^{t_2} y_{p_j}^{y_{p_j}})^{h(m,R,ProxyID)} \bmod q.$$

- Security: The security analysis of the scheme is weak. With some heuristic arguments the authors proved the security strength of the scheme.

5.1.9 Herranz and Saez (2004)

Proposed a distributed proxy signature scheme. The scheme extended the work in [3] to the scenario of fully distributed proxy signature schemes.

Assumption: DLP is hard.

- Generation of keys: Let $E = \{P^{(1)}, P^{((2)}, \cdots, P^{(n)}\}$ be a distributed entity formed by n participants. There is an *access structure* $\Gamma \subset 2^E$, which is formed by those subsets of participants which are authorized to perform the secret task. The access structure must be monotone increasing; that is, if $A_1 \in \Gamma$ is authorized and $A_1 \subset A_2 \subset E$, then A_2 must be authorized. The joint generation of discrete logarithm keys is as follows:
  - Each participant $P^{(l)} \in E$ obtains a secret value $x^{(l)} \in \mathbb{Z}_{q-1}$. The values $\{x^{(l)}\}_{P^{(l)} \in E}$ form a sharing of the secret key $x \in \mathbb{Z}_{q-1}$, according to some linear secret sharing scheme realizing the access structure Γ. The corresponding public key $y = g^x \bmod q$ is made public, along with other values (commitments) which ensure the robustness of the protocol.
  - The fully distributed *triple* Schnorr proxy signature scheme is generated in a similar way of the Boldyreva et al.'s scheme [3].
- Security: The scheme is as secure as Kim et al.'s PDW scheme [38].

5.1.10 Malkin, Obana and Yung (2004)

Presented a formal model for fully hierarchical proxy signatures with warrant that supports chains of several levels of delegation.

Assumption: DLP is hard.

- The signers' selects private key x_i and computes public key $y_i \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_i)$.
- Delegation capability generation: This is an interactive process between the designator and proxy signer. It takes public keys of a designator $y_{i_{L-1}}$ and a proxy

signer y_{i_L}, the signing key of which the designator delegates its signing right (i.e., the signing key is either a signing key $x_{i_{L-1}}$ or a proxy key $\sigma_{i_o \cdots \rightarrow i_{L-1}}$ depending on whether i_{L-1} is original signer or proxy signer), a warrant up to previous delegation W_{L-1} and a warrant ω_L set in current delegation as inputs; outputs delegation rights.

- Proxy key generation: It takes public keys of a designator $y_{i_{L-1}}$ and a proxy signer y_{i_l}, the private key of the proxy signer x_{i_L} as inputs and outputs a proxy key $\sigma_{i_o \cdots \rightarrow i_L}$ and a warrant ω.
- Proxy signature generation: The proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, \sigma_{i_o \cdots \rightarrow i_L}, (m, \omega)).$$

- Proxy signature verification: The verifier accepts the proxy signature if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_{i_o}, \sigma_p, (m, \omega)).$$

- Security: The scheme formalizes a model of fully hierarchical proxy signature, which is a probably secure model to the best of our knowledge.

5.1.11 Lu and Huang (2006)

Proposed a proxy signature scheme using time-stamping service for validating delegation service at the verifier.

Assumption: DLP is hard.

- Alice picks a private key x_o and generates public key $y_o \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_o)$.
- Bob picks a private key x_p and generates public key $y_p \leftarrow \mathcal{KG}_{dlp}(\textbf{params-dlp}, x_p)$.
- Delegation capability generation: Alice chooses a random number $k_o \in \mathbb{Z}^*_{q-1}$ and computes $r_o = g^{k_o} \bmod q$. Then she computes $\sigma_o \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_o, r_o), x_o, \omega)$.
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, y_o, r_o, \sigma_o, \omega).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{dlp}(\textbf{params-dlp}, \sigma_o, x_p, \textit{public-parameters})$, and $y'_p \leftarrow g^{\rho_p}$.
- Proxy signature generation: Firstly, Alice sends ω to a time-stamping service (TSS) for a time-stamp. The TSS generates $t_B \leftarrow h(n, \omega, t_{B-1}, t_{f(B)})$ and sends it back to Alice, where n is the group size. Then Alice makes the (ω, t_B) to the public. Secondly, Bob sends a message m to the TSS and requests a time-stamp. The TSS generates a time-stamp $t_n \leftarrow h(n, m, t_{n-1}, t_{f(n)})$ and sends it back to Bob. Finally, the proxy signature on message m is computed as

$$\sigma_p \leftarrow \mathcal{S}_{dlp}(\textbf{params-dlp}, (k_p, r_p), \rho_p, m, t_n).$$

- Proxy signature verification: The verifier accepts the proxy signature if and only if

$$Valid \leftarrow \mathcal{V}_{dlp}(\textbf{params-dlp}, (y_o, y_p), \sigma_p, (m, t_n, \omega)).$$

- Security: The scheme's security relies on DLP. The use of time-stamp provides a mechanism for the delegation expiry or revoking by Alice, if she desires to do so.

Table 2: Conventions and notation for RSA-based proxy signature schemes

Alice	Original signer
Bob	Proxy signer
N_o, N_p	RSA Modulus for Alice and Bob, respectively
y_o	Public key of Alice, where $1 < y_o < \phi(N_o)$
y_p	Public key of Bob, where $1 < y_p < \phi(N_p)$
x_o	Private key of Alice, where $x_o y_o \equiv 1 \bmod \phi(N_o)$
x_p	Private key of Bob, where $x_p y_p \equiv 1 \bmod \phi(N_p)$
ω	A warrant
$h(.)$	A collision-resistant one-way hash function

5.2 RSA-based Proxy Signature

5.2.1 Okamoto, Tada and Okamoto (1999)

Proposed a scheme that reduces the computation and storage cost during the protocol execution, and the protocol is suitable for implementation on smart cards.

Assumption: IFP is hard and smart card is a tamper resistant device.

- Alice picks a public key y_o and generates private key $x_o \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_o)$.
- Delegation capability generation: Alice computes $\sigma_o \leftarrow \mathcal{S}_{rsa}(\textbf{params-rsa}, x_o, (\omega, I_p))$ where I_p denote the limit of money which she can spend.
- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{rsa}(\textbf{params-rsa}, y_o, \sigma_o, (\omega, I_p)).$$

- Proxy signature generation: To sign a message m, Bob generates a random number $k_p \in \mathbb{Z}^*_N$, and computes

$$\begin{aligned} r &= g^{k_p h(m)} \sigma_o \bmod N_o \\ s &= g^{-y_o k_p} \bmod N_o. \end{aligned}$$

- The proxy signature of message m is the tuple $(m, (r, s), I_p)$.
- Proxy signature verification: The verifier checks whether $(ID_p, \omega) = h(I_p) r^{y_o} s^{h(m)} \bmod N_o$. If it does, the verifier accepts it as a valid proxy signature. Otherwise, rejects it.
- Security: The scheme has weak security as it is designed as a proxy-unprotected scheme, where Alice can frame Bob by signing the message and later claim that Bob has signed the message.

5.2.2 Lee, Kim and Kim (2001b)

A mobile agent is constructed in the scheme using non-designated proxy signature which represents both the original signer's (customer) and the proxy signer's (remote server) signatures.

Assumption: IFP is hard.

- Alice picks a public key y_o and generates private key $x_o \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_o)$.
- The mobile agent (proxy signer) chooses a public key y_p and generates private key

$$x_p \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_p).$$

- Delegation capability generation: Alice creates $\sigma_o \leftarrow \mathcal{S}_{rsa}(\textbf{params-rsa}, x_o, (AliceID, req))$, where req is the customer requirements for purchase such as price range, date, delivery requirements, etc.
- Delegation capability verification: The mobile agent accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{rsa}(\textbf{params-rsa}, y_o, \sigma_o, (AliceID, req)).$$

- Proxy signature generation: Let BID be the agent's bid information which conforms to req. The agent (remote server) tries to sell the product to Alice. The remote server computes

$$\begin{aligned} x &= h(AliceID, req, AgentID, BID)^{x_p} \bmod N_p \\ y &= h(AliceID, req)^x \bmod N_o, \text{ and} \\ z &= \sigma_o^x \bmod N_o. \end{aligned}$$

  Then sends the tuple ($AliceID$, req, $AgentID$, BID, x, y, z) to the mobile agent and the agent will get back to Alice with this tuple as a receipt of the purchase.
- Proxy signature verification: Alice receives ($AliceID$, req, $AgentID$, BID, x, y, z) from the mobile agent, then she can verify the validity of the purchase by the following:
  - Whether $h(AliceID, req, AgentID, BID) = x^{y_p} \bmod N_p$.
  - Whether $y = h(AliceID, req)^x \bmod N_o$.
  - Whether $y = z^{y_o} \bmod N_o$.
  - Whether $BID \in \{req\}$.
- Security: Wang et al. [80] showed that the scheme is insecure and inefficient.

5.2.3 Shao (2003)

Proposed a proxy signature scheme based on the factoring problem, which combines the RSA signature scheme and the Guillou and Quisquater [24] signature scheme.

Assumption: IFP is hard and Guillou-Quisquater signature is secure.

- Alice picks a public key y_o and generates private key $x_o \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_o)$.
- Bob picks a public key y_p and generates private key $x_p \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_p)$.
- Delegation capability generation: Alice computes proxy key $v = h(\omega, ProxyID)^{-x_o} \bmod N_o$, $u = \lfloor v/N_p \rfloor$ and $z = v^{y_p} \bmod N_p$. The delegation is the tuple (ω, z, u).
- Delegation capability verification: Bob recovers $v = u \times N_p + (z^{x_p} \bmod N_p)$.
- Proxy signature generation: Let m be the message to be signed by Bob. Bob does the following:
  - Randomly chooses an integer $t \in [1, N_o]$ and computes $r = t^{y_o} \bmod N_o$.
  - Compute $k = h(m, r)$ and $x = k^{x_p} \bmod N_p$.
  - Compute $y = tv^k \bmod N_o$. The proxy signature on message m is $(m, \omega, x, y, ProxyID)$.
- Proxy signature verification: The verifier checks the following:
  - Compute $k' = x^{y_p} \bmod N_p$.
  - Compute $r' = y^{y_o} h(\omega, ProxyID)^{k'} \bmod N_o$.
  - Check whether $k' = h(m, r')$.
- Security: The security of the scheme is based on Guillou-Quisquater signature scheme [24]. However, there is no formal security proof of it.

5.2.4 Das, Saxena and Gulati (2004)

Proposed a proxy signature scheme that provides effective proxy revocation mechanism.

Assumption: IFP is hard. In addition to Alice and Bob, a trusted server (TS) is a participant in the scheme for time stamp issuance.

- Alice picks a public key y_o and generates private key $x_o \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_o)$.

- Bob picks a public key y_p and generates private key $x_p \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_p)$.
- TS chooses a public key y_s and generates private key $x_s \leftarrow \mathcal{KG}_{rsa}(\textbf{params-rsa}, y_s)$.
- Delegation capability generation: Alice computes the delegation capability σ_o as

$$\sigma_o \leftarrow \mathcal{S}_{rsa}(\textbf{params-rsa}, x_o, (\omega, y_p, y_s)).$$

- Delegation capability verification: Bob accepts σ_o if and only if

$$Valid \leftarrow \mathcal{V}_{rsa}(\textbf{params-rsa}, y_o, \sigma_o, (\omega, y_p, y_s)).$$

- Proxy signature generation: Let m be the message to be signed. Bob requests a time stamp to the TS and sends (R, m, y_o, y_p), where $R \leftarrow \mathcal{S}_{rsa}(\textbf{params-rsa}, x_p, (m, \omega, y_o, y_p))$. The TS verifies whether $Valid \leftarrow \mathcal{V}_{rsa}(\textbf{params-rsa}, y_p, R, (m, \omega, y_o, y_p))$. If it holds, the TS ascertain the following conditions are true before the time stamp is issued:
  - Alice's public key y_o is not in the public revocation list maintained by the TS.
  - ω is not expired.

  Now, TS computes $T_m \leftarrow \mathcal{S}_{rsa}(\textbf{params-rsa}, x_s, (m, \omega, y_o, y_p, T))$, where T denotes a time stamp. Then, TS sends (T_m, T) to Bob over a public channel.

  Bob accepts T if and only if

$$Valid \leftarrow \mathcal{V}_{rsa}(\textbf{params-rsa}, y_s, T_m, (m, \omega, y_o, y_p, T)).$$

  If it holds, Bob generates proxy signature as $\sigma_p = (h(m, \omega, y_o, y_s, T) \oplus \sigma_o)^{x_p} \bmod N_p$, otherwise rejects the time stamp and makes another request to the TS. The proxy signature of message m is $(m, \omega, T, T_m, \sigma_p)$.
- Proxy signature verification: The verifier checks the following to validate a proxy signature:
  - Whether $Valid \leftarrow \mathcal{V}_{rsa}(\textbf{params-rsa}, y_s, T_m, (m, \omega, y_o, y_p, T))$. If it holds, the verifier is assured that the time stamp in the signed message is correct. Otherwise, he rejects the signed message.
  - Whether $h(\omega, y_p, y_s) = (\sigma_p^{y_p} \bmod N_p \oplus h(m, \omega, y_o, y_s, T))^{y_o} \bmod N_o$. If it holds, he accepts the signed message. Otherwise, he rejects it.
- Security: The scheme provides an effective proxy revocation mechanism. The scheme is secure on the assumption that IFP is hard. However, the scheme does not work when $N_p > N_o$, but it is a valid assumption because typically the proxy signer key strength should not be greater than the original signer key strength.

5.2.5 Zhou, Cao and Chai (2005)

Proposed a warrant-based proxy signature scheme, where security is based on IFP. They used improved Rabin's signature scheme [62] for their scheme.

Assumption: IFP is hard.

- Alice picks public key (N_o, a_o) and private key (p_o, q_o), where p_o and q_o are two large primes, N_o is the product of these two primes and $a_o \in \mathbb{Z}^*_{N_o}$ is Jacobi symbol satisfying $(\frac{a_o}{N_o})$ = -1.
- Bob picks public key (N_p, a_p) and private key (p_p, q_p), where p_p and q_p are two large primes, N_p is the product of these two primes and $a_p \in \mathbb{Z}^*_{N_p}$ is Jacobi symbol satisfying $(\frac{a_p}{N_p})$ = -1.
- Delegation capability generation: Alice signs on warrant and computes delegation power σ_o as $\sigma_o \leftarrow \mathcal{S}_{rabin}(p_o, q_o, a_o, \omega, N_o)$. We note that the signature process is done by Rabin's [62] signature generation phase, and we term it as $\mathcal{S}_{rabin}$.
- Delegation capability verification: Bob verifies whether $Valid \leftarrow \mathcal{V}_{rabin}(\sigma_o, \omega, a_o, N_o)$. We note that the verification process is done by Rabin's [62] signature verification phase, and we term it as $\mathcal{V}_{rabin}$.
- Proxy signature generation: Let m be the message to be signed by Bob. Bob creates his signature on m as $\sigma_p \leftarrow \mathcal{S}_{rabin}(p_p, q_p, a_p, \sigma_o, N_p)$.
- Proxy signature verification: The verifier accepts the proxy signature σ_p if and only if $Valid \leftarrow \mathcal{V}_{rabin}(\sigma_p, \sigma_o, \omega, a_o, a_p, N_o, N_p)$.
- Security: The scheme is secure on the assumption that IFP is hard. The authors also proved the security strength of the scheme under random oracle model.

5.3 ECDSA-based Proxy Signature

5.3.1 Chen, Chung and Huang (2003)

Proposed a scheme for proxy multi-signature based on elliptic curve cryptosystem that reduces high computational overheads of Sun's scheme [71].

Assumption: ECDLP is hard.

- Let $B = (x_B, y_B)$ be a point in $E(F_q)$ for a large prime q, the order of B is assumed as t. For each $1 \le i \le n$, the original signer secretly selects a random number $1 \le d_i \le t-1$ as her private key and computes the corresponding public key $Q_i = d_i \times B = (x_{Q_i}, y_{Q_i})$.
- The proxy signer selects a private key $1 \le d_p \le t-1$ and computes corresponding public key $Q_p = d_p \times = (x_{Q_p}, y_{Q_p})$.

- Delegation capability generation: For each $1 \leq i \leq n$, the original signer A_i selects a random number $1 \leq k_i \leq t-1$, computes $r_i = k_i \times B = (x_{r_i}, y_{r_i})$ and $s_i = x_i \cdot x_{Q_i} \cdot h(\omega, r_i) k_i \bmod t$.

- Delegation capability verification and proxy key generation: For each $1 \leq i \leq n$, the proxy signer computes $U_i = (x_{Q_i} \cdot h(\omega, r_i) \bmod t) \times Q_i - s_i \times B = (x_{U_i}, y_{U_i})$ using (ω, r_i, s_i). If $x_{U_i} = x_{r_i} \bmod t$, the proxy signer accepts s_i as a valid delegation of signing right; otherwise, he rejects it. If the proxy signer validates all (ω, r_i, s_i) in which $1 \leq i \leq n$, s/he then computes $d = d_p \cdot x_{Q_p} + \sum_{i=1}^{n} s_i \bmod t$ as a valid proxy key.

- Proxy signature generation: When the proxy signer signs a message m for $A_1, \cdots, A_n$, he executes the signing operation of a designated signature scheme using the signing key d. The resulting proxy signature is the tuple $(m, \sigma_{s_p}(m), r_1, r_2, \cdots, r_n, \omega)$.

- Proxy signature verification: The verifier computes the proxy public key Q corresponding to the proxy key s_p for verifying the proxy signature by the designated signature scheme: $Q = x_{Q_p} \times Q_p + (x_{Q_1} \cdot h(\omega, r_1) \bmod t \times Q_1 + \cdots + (x_{Q_n} \cdot h(\omega, r_n) \bmod t \times Q_n (r_1 + \cdots + r_n)$. With the newly generated proxy public key Q, the verifier confirms the validity of $\sigma_{s_p}(m)$ by validating the verification equation of the designated signature scheme.

- Security: The authors did not consider any security model for their scheme, instead, a heuristic security analysis is given to safeguard the scheme.

5.4 Pairing-based Proxy Signature

5.4.1 Zhang, Safavi-Naini and Lin (2003)

Proposed an identity based proxy signature based on Hess's ID-based signature.

Assumption: WDHP is hard.

- Alice computes her public key $y_o = H(ID_o)$, where ID_o is her identity. Then, Alice obtains her private key $x_o \leftarrow \mathcal{KG}_{cdhp}(\textbf{params-cdhp}, y_o)$ from KGC.

- Bob computes his public key $y_p = H(ID_p)$, where ID_p is his identity. Then, Bob obtains his private key $x_p \leftarrow \mathcal{KG}_{cdhp}(\textbf{params-cdhp}, y_p)$ from KGC.

- Delegation capability generation: Alice computes $\sigma_o \leftarrow \mathcal{S}_{cdhp}(\textbf{params-cdhp}, (k_o, r_o, c_o), x_o, \omega)$.
  Delegation capability verification: Bob accepts σ_o if and only if

  $$Valid \leftarrow \mathcal{V}_{cdhp}(\textbf{params-cdhp}, y_o, \sigma_o, (c_o, \omega)).$$

- Proxy key generation: Bob computes proxy key $\rho_p \leftarrow \texttt{PKeyGen}_{cdhp}(\textbf{params-cdhp}, \sigma_o, c_o, x_p)$.

- Proxy signature generation: Bob picks a random number $k_p \in \mathbb{Z}_{q-1}^*$, computes

  $$\begin{aligned} r_p &= \hat{e}(P,P)^{k_p}, \\ c_p &= h(m, r_p) \text{ and} \\ \sigma_p &\leftarrow \mathcal{S}_{cdhp}(\textbf{params-cdhp}, (k_p, c_p), \rho_p, m). \end{aligned}$$

- Proxy signature verification: The verifier accepts the proxy signature if and only if

  $$Valid \leftarrow \mathcal{V}_{cdhp}(\textbf{params-cdhp}, (y_o, y_p), \sigma_p, (c_p, m, \omega)).$$

- Security: Only heuristic security analysis is given to safeguard the scheme.

5.4.2 Chen, Zhang and Kim (2003)

Proposed a multi-proxy signature scheme, where Alice delegates her signing capability to l proxy signers.

Assumption: CDHP is hard.

- Alice computes her public key $y_o = H(ID_o)$, where ID_o is her identity. Then, Alice obtains her private key $x_o \leftarrow \mathcal{KG}_{cdhp}(\textbf{params-cdhp}, y_o)$ from KGC.

- Each proxy signer computes his public key $y_{p_i} = H(ID_{p_i})$, where ID_{p_i} is his identity. Then, Each proxy signer obtains his private key $x_{p_i} \leftarrow \mathcal{KG}_{cdhp}(\textbf{params-cdhp}, y_{p_i})$ from KGC.

- Delegation capability generation: Alice picks a random number $k_o \in \mathbb{Z}_{q-1}^*$, computes $r_o = \hat{e}(P,P)^{k_o}$, $c_o = h(\omega, r_o)$, $\sigma_o \leftarrow \mathcal{S}_{cdhp}(\textbf{params-cdhp}, (k_o, r_o, c_o), x_o, \omega)$.

- Delegation capability verification: Each proxy signer accepts σ_o if and only if

  $$(k_p, c_p), \leftarrow \mathcal{V}_{cdhp}(\textbf{params-cdhp}, y_o, \sigma_o, (c_o, \omega)).$$

- Proxy key generation: Each proxy signer computes proxy key as

  $$\rho_{p_i} \leftarrow \texttt{PKeyGen}_{cdhp}(\textbf{params-cdhp}, \sigma_o, c_o, x_{p_i}).$$

- Proxy signature generation: Each proxy signer performs the following operations to sign a message m:
  - Pick randomly $k_{p_i} \in \mathbb{Z}_{q-1}^*$, computes $r_{p_i} = \hat{e}(P,P)^{k_{p_i}}$ and broadcasts r_{p_i} to the remaining $l-1$ proxy signers.
  - Compute $r_p = \prod_{i=1}^{l} r_{p_i}$ and $c_p = h(m, r_p)$, $\sigma_{p_i} = c_p \rho_{p_i} + k_{p_i} P$. Then, send σ_{p_i} to a designated clerk (one of the proxy signers).
  - The clerk verifies the individual proxy signature and computes $\sigma_p = \sum_{i=1}^{l} \sigma_{p_i}$. The proxy signature of message m is the tuple $(m, \omega, r_o, c_p, \sigma_p)$.

Table 3: Conventions and notation for pairings-based proxy signature schemes

Alice	Original signer
Bob	Proxy signer
G_1	A cyclic additive group, whose order is prime q
G_2	A cyclic multiplicative group of the same order q
P	A generator of G_1
$\hat{e}$	A bilinear pairing map
$H(\cdot)$	Map-to-Point function
$h(\cdot)$	Collision-resistant one-way hash function
s	PKG's master-key
Pub_{KGC}	KGC's public key, $Pub_{KGC} = sP$
y_o, x_o	Alice's public key and private key, respectively, $y_o = H(ID_o)$
y_p, x_p	Bob's public key and private key, respectively, $y_p = H(ID_p)$

- Proxy signature verification: The verifier accepts the proxy signature of message m if and only if $c_p = h(m, \hat{e}(\sigma_p, P)(\hat{e}(\sum_{i=1}^{l}(y_o + y_{p_i}, Pub_{KGC})^{h(\omega, r_o)} \cdot r_o^l)^{-c_p}$.
- Security: No formal security proof is considered to safeguard the scheme.

5.4.3 Xu, Zhang and Feng (2004)

Formalized a notion of security for ID-based proxy signature schemes and presented a proxy signature scheme.

Assumption: CDHP is hard.

- Alice computes her public key $y_o = H(ID_o)$, where ID_o is her identity. Then, Alice obtains her private key $x_o \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_o) from KGC.
- Bob computes his public key $y_p = H(ID_p)$, where ID_p is his identity. Then, Bob obtains his private key $x_p \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_p) from KGC.
- Delegation capability generation: Alice picks $k_o \in \mathbb{Z}_q^*$, computes $r_o = k_o P$, $C_o = H_o(ID_o, \omega, r_o)$ and $\sigma_o = x_o + k_o C_o$, where
$$H_o : \{0,1\}^* \times \{0,1\}^* \times G_1 \rightarrow G_1.$$
- Delegation capability verification: Bob accepts σ_o if and only if
$$\hat{e}(\sigma_o, P) = \hat{e}(Pub_{KGC}, y_o)\hat{e}(r_o, C_o).$$
- Proxy key generation: Bob computes proxy key $\rho_p = h_1(ID_o, ID_p, \omega, r_o)x_p + \sigma_o$, where
$$h_1 : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*.$$
- Proxy signature generation: To sign a message m, Bob does the following:
  - Picks $k_p \in \mathbb{Z}_{q-1}^*$, computes $r_p = k_p P$ and then puts $C_p = H_o(ID_p, m, r_p)$.
  - Computes $\sigma_p = \rho_p + k_p C_p$. The proxy signature of message m is the tuple $((r_o, r_p), \sigma_p, (\omega, m))$.
- Proxy signature verification: The verifier accepts the proxy signature of message m iff $\hat{e}(\sigma_p, P) = \hat{e}(Pub_{KGC}, y_p)^{h_1(ID_o, ID_p, \omega, r_o)}\hat{e}(Pub_{KGC}, y_o)\hat{e}(r_p, C_p)\hat{e}(r_o, C_o)$.
- Security: Security of the scheme is based on the CDHP in the random oracle model. But, the scheme takes large computational cost.

5.4.4 Zhang, Safavi-Naini and Susilo (2004)

Proposed a proxy signature scheme based on a short signature scheme.

Assumption: CDHP is hard.

- Alice computes her public key $y_o = H(ID_o)$, where ID_o is her identity. Then, Alice obtains her private key $x_o \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_o) from KGC.
- Bob computes his public key $y_p = H(ID_p)$, where ID_p is his identity. Then, Bob obtains his private key $x_p \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_p) from KGC.
- Delegation capability generation: Alice computes $\sigma_o = (s_o + h(\omega))^{-1} y_p$.
- Delegation capability verification: Bob accepts σ_o if and only if
$$\hat{e}(h(\omega)P + y_o, \sigma_o) = \hat{e}(P, y_p).$$
- Proxy key generation: Bob computes $\rho_p = x_p \sigma_o$.
- Proxy signature generation: To sign a message m, Bob does the following.
  - Chooses a random number $r \in \mathbb{Z}_{q-1}^*$ and computes $U = r \cdot (h(\omega)P + y_o)$.
  - Computes $t = H_2(m, U)$ and $\sigma_p = (t + r)^{-1}\rho_p$, where $H_2 : \{0,1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$. The proxy signature of message m is (U, σ_p, ω).

- Proxy signature verification: The verifier verifies whether

$$\hat{e}(U + H_2(m, U)(h(\omega)P + y_o), \sigma_p) = \hat{e}(y_p, y_p).$$

- Security: Security of the scheme is based on CDHP in the random oracle model.

5.4.5 Lu, Cao and Dong (2006)

Proposed a designated verifier proxy signature scheme.

Assumption: CDHP is hard.

- Alice computes her public key $y_o = H(ID_o)$, where ID_o is her identity. Then, Alice obtains her private key $x_o \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_o) from KGC.
- Bob computes his public key $y_p = H(ID_p)$, where ID_p is his identity. Then, Bob obtains his private key $x_p \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_p) from KGC.
- Martin (a designated verifier) computes his public key $y_m = H(ID_m)$, where ID_m is his identity. Then, Martin obtains his private key $x_m \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_m) from KGC.
- Delegation capability generation: Alice generates delegation capability σ_o as $\sigma_o = x_o H(\omega)$.
- Delegation capability verification: Bob accepts σ_o if and only if $\hat{e}(\sigma_o, P) = \hat{e}(H(\omega), y_o)$.
- Proxy key generation: Bob computes $\rho_p = \sigma_o + x_p H(\omega)$.
- Proxy signature generation: To generate a designated verifier proxy signature for Martin on message m, Bob does the following:
  - Picks $k_p \in \mathbb{Z}^*_{q-1}$ and computes $r_p = k_p y_m$.
  - Computes $\sigma_p = k_p(y_o + y_p) - H_2(m, r_p) \cdot \rho_p$, where $H_2 : \{0,1\}^* \times G_1 \rightarrow \mathbb{Z}^*_q$.
  - The proxy signature of message m is the tuple $(r_p, \sigma_p, (\omega, m))$.
- Proxy signature verification: The verifier accepts the proxy signature of message m if and only if $\hat{e}(y_o + y_p, r'_p) = \hat{e}(\sigma_p, P) \cdot \hat{e}(H(\omega), y_o + y_p)^{H_2(m, r_p)}$, where $r'_p = \frac{1}{x_m} \cdot r_p$.
- Security: Security of the scheme is based on the CDHP in the random oracle model. But, the scheme requires secure channel for proxy delivery.

5.4.6 Das, Saxena and Phatak (2007)

Proposed a proxy signature scheme based on Hess signature scheme that provides effective proxy revocation mechanism and avoids key escrow problem.

Assumption: CDHP is hard.

- Alice computes her public key $y_o = H(ID_o)$, where ID_o is her identity. Then, Alice generates her private key $x_o \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_o).
- Bob computes his public key $y_p = H(ID_p)$, where ID_p is his identity. Then, Bob generates his private key $x_p \leftarrow \mathcal{KG}_{cdhp}$(**params-cdhp**, y_p).
- Delegation capability generation: Alice computes $\sigma_o = (s_o + b_o H'(\omega, y_o, y_p)$, and $\psi_o = b_o P$. Here, b_o is secret to Alice only and $H' : \{0,1\}^* \times G_1 \times G_1 \rightarrow G_1$.
- Delegation capability verification: Bob accepts σ_o if and only if

$$\hat{e}(s_o, P) = \hat{e}(\psi_o, H'(\omega, y_o, y_p)) \cdot \hat{e}(y_o, Reg_o),$$

  where $Reg_o = sb_o P$, registration token published by the KGC.
- Proxy key generation: Bob computes $\rho_p = s_o + s_p + b_p H'(\omega, y_o, y_p)$. Here, b_p is secret to the proxy signer only.
- Proxy signature generation: To sign a message m, Bob does the following.
  - Selects a random $r \in \mathbb{Z}^*_q$ and compute $R = rP$.
  - Computes $a = h(m, R, y_p)$ and $\psi_p = b_p P$, where $h : \{0,1\}^* \times G_1 \times G_1 \rightarrow \{0,1\}^*$.
  - Computes $\sigma_p = (r + a)^{-1} \rho_p$. The proxy signature of message m is $(\omega, m, R, \sigma_p, \psi_o, \psi_p, y_o, y_p)$.
- Proxy signature verification: The proxy signature is valid if and only if

$$\begin{aligned} & \hat{e}(R + h(m, R, y_p)P, \sigma_p) \\ = \; & \hat{e}(\psi_o + \psi_p, H'(\omega, y_o, y_p)) \cdot \hat{e}(y_o, Reg_o) \cdot \hat{e}(y_p, Reg_p). \end{aligned}$$

- Security: The scheme is secure and does not require secure channel in key issuance stage.

6 Concluding Remarks

We have reviewed a few seminal works on proxy signatures with respect to different security assumptions. In order to give a concise picture of the schemes highlighting the important features and security aspects at a glance, we compare them in the following tables. The Table 4 depicts the DLP-based schemes, Table 5 depicts the RSA-based schemes, and Table 6 depicts the Pairing-based schemes. We note that the computational complexity of the schemes in a same table more or less similar, as their underlying security is based on the same cryptographic primitive. It is observed that many times, a paper typically breaks a previous scheme and proposes a new one, which someone breaks later and, in turn, proposes a new one, and so on. Most of such work, though quite important and useful, essentially provides an incremental advance to the same basic theme. Consequently, we believe

Table 4: Comparisons of the DLP-based proxy signatures

Features → *Schemes* ↓	*Secure*	*Secure Channel*	*Proxy Revocation*	*Remarks*
Mambo et al., 1996	No	Yes	Partial	Unlimited delegation and misuse of delegation
Kim et al., 1997	Yes	No	No	Secure
Zhang, 1997a	No	Yes	No	Insecure
Petersen and Hoster, 1997	No	No	No	Insecure
Sun et al., 1999	No	Yes	No	Suffers from Coalition attacks
Lee et al., 2001b	No	No	No	Suffers from Transferring, Forgery attacks
Boldyreva et al., 2003	Yes	No	No	Based on Kim et al., 1997 and formalizes the security notion, but unlimited delegation
Li et al., 2003	Yes	No	No	No formal security analysis
Herranz and Saez, 2004	Yes	No	No	Distributed proxy signature, ideas based on Boldyreva et al., 2003a
Malkin et al., 2004	Yes	No	No	Secure, based on Kim et al., 1997, hierarchical delegation
Lu and Huang, 2006	No	No	No	Suffers from Transferring, Forgery attacks

Table 5: Comparisons of the RSA-based proxy signatures

Features → *Schemes* ↓	*Secure*	*Secure Channel*	*Proxy Revocation*	*Remarks*
Okamoto et al., 1999	Yes	Yes	No	Does not provide strong unforgeability and prevention of misuse
Lee at al., 2001b	No	Yes	No	Insecure
Shao, 2003	No	Yes	No	No formal security proof
Das et al., 2004	Yes	No	Yes	$N_p < N_o$
Zhou et al., 2005	Yes	No	No	Considers random oracle model

Table 6: Comparisons of the pairing-based proxy signatures

Features → *Schemes* ↓	*Key Escrow*	*Secure Channel*	*Proxy Revocation*	*Remarks*
Zhang et al., 2003	Yes	Yes	No	No formal security proof, suffers from key escrow, needs secure channel
Chen et al., 2003	Yes	Yes	No	No formal security proof, suffers from key escrow, needs secure channel
Xu et al., 2004	Yes	Yes	No	Takes high computation cost, suffers from key escrow, needs secure channel
Zhang et al., 2004	Yes	Yes	No	Suffers from key escrow, needs secure channel
Lu et al., 2006	Yes	Yes	No	Suffers from key escrow, needs secure channel
Das et al., 2007	No	No	Yes	Secure, no key escrow, no secure channel

it is not worth to measure a mathematical figure if the scheme is already found insecure. Instead, we summarize the merits and limitations of the schemes. Finally, we extract a few schemes from each category, which are seemed to be computationally efficient and secure.

From these tables, it is clear to select some schemes which seem to be secure and efficient. The Kim et al. [38] proposed a seminal work which invites many other proposal on the same basic theme and few of them later found insecure. To the best of authors knowledge, the Kim et al.'s scheme is secure and efficient ones. Although Malkin et al. [52] is also secure and efficient; however, it is based on Kim et al.'s scheme. Lu and Huang [49] is also found secure and provides proxy revocation mechanism. The Zhou et al.'s scheme [89] is the only candidate from the RSA-based approach, which seems to be efficient and secure. And, from the pairing-based approach, Das et al. [16] has shown potential for the security strength compared to other schemes, though because of pairing operation it takes more computational cost than the DLP and RSA-based schemes.

We also tried to explore if there are any real implementation of various proposed proxy signatures. For this, we contacted individual author of some of the papers over email, to learn whether the scheme is used in any real-world applications? Nonetheless, we have not identified a scheme which is being used in practical application. In fact, the responses from several authors indicated that they were not aware of such applications which use their scheme.

In conclusion, we believe that the actual deployment of proxy signatures is yet to start in a big way. However, as and when this happens, the research work being carried out will certainly provide practically usable implementations.

References

[1] A. K. Awasthi, and S. Lal, "A multi-proxy signature scheme for partial delegation with warrant," 2005. (http://arxiv.org/pdf/cs.CR/0504093)

[2] H. Bao, Z. Cao and, S. Wang, "Identity-based threshold proxy signature scheme with known signers," *Proceedings of Theory and Applications of Models of Computation*, LNCS 3959, pp. 538-546, Springer-Verlag, 2006.

[3] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure signature schemes for delegation of signing rights," 2003. (http://eprint.iacr.org/2003/96/)

[4] D. Boneh, and M. Franklin, "Identity-based encryption from the Weil pairing," *Proceedings of Crypto' 01*, LNCS 2139, pp. 213-229, Springer-Verlag, 2001.

[5] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," *Proceedings of Asiacrypt' 01*, LNCS 2248, pp. 514-532, Springer-Verlag, 2001.

[6] C. Boyd, and A. Mathuria, "Protocols for authentication and key establishment," Springer-Verlag, 2003.

[7] F. Cao, and Z. Cao, "A proxy-protected signature scheme based on finite automaton," *Proceedings of International Multi-Symposiums on Computer and Computational Sciences*, pp. 48-55, IEEE Computer Society, 2006

[8] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption zscheme," *Proceedings of Eurocrypt' 03*, LNCS 2656, pp. 255-271, Springer-Verlag, 2003.

[9] T. S. Chen, Y. F. Chung, and G. S. Huang, "Efficient proxy multi-signature schemes based on the elliptic curve cryptosystem," *Computers & Security*, vol. 22, no. 6, pp. 527-534, 2003.

[10] T. S. Chen, Y. F. Chung, and K. H. Huang, "A traceable proxy multi-signature acheme based on the elliptic surve cryptosystem," *Applied Mathematics and Computation*, vol. 159, no. 1, pp. 137-145, 2004.

[11] X. Chen, F. Zhang, and K. Kim, "ID-based multi-proxy signature and blind multi-signature from bilinear pairings," *Proceedings of KIISC' 03*, pp. 11-19, 2003.

[12] C. Cocks, "An identity based encryption scheme based on quadratic residues," *Cryptography and Coding*, LNCS 2260, pp. 360-363, Springer-Verlag, 2001.

[13] M. L. Das, A. Saxena, and V. P. Gulati, "Proxy signatures using partial delegation with warrant," *Proceedings of International Conference on Number Theory for Secure Communication*, pp. 152-154, 2003.

[14] M. L. Das, A. Saxena, and V. P. Gulati, "Security analysis of Lal and Awasthi's proxy signature schemes", 2003. (http://eprint.iacr.org/2003/263/)

[15] M. L. Das, A. Saxena, and V. P. Gulati, "An efficient proxy signature scheme with revocation," *Informatica*, vol. 15, no. 4, pp. 455-464, 2004.

[16] M. L. Das, A. Saxena, and D. B. Phatak, "A proxy signature scheme with revocation using bilinear pairings," *International Journal of Network Security*, vol. 4, no. 3, pp. 312-317, 2007.

[17] W. Diffie, and M. E. Hellman, "New directions in cryptography.", *IEEE Transactions on Information Theory*, vol. IT-22, pp. 644-654, 1976.

[18] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. IT-31, no. 4, pp. 469-472, 1985.

[19] G. Frey, and H. Ruck, "A remark roncerning m-divisibility and the discrete logarithm in the divisor class group of curves," *Mathematics of Computation*, vol. 62, pp. 865-874, 1994.

[20] M. Gasser, A. Goldstein, C. Kaufman, and B. Lampson, "The digital distributed system security architecture," *Proceedings of National Computer Security Conference*, pp. 305-319, 1989.

[21] R. Gennaro. S. Jarecki, H. Krawczyk, and T. Rabi, "Robust threshold DSS signatures," *Proceedings of Eurocrypt' 96*, LNCS 1070, pp. 354-371, Springer-Verlag, 1996.

[22] H. Ghodosi, and J. Pieprzyk, "Repudiation of cheating and non-repudiation of Zhang's proxy signature schemes," *Proceedings of Australasian Conference on Information Security and Privacy (ACISP' 99)*, LNCS 1587, pp. 129-134, Springer-Verlag, 1999.

[23] S. Goldwasser, S. Micali, and R. Rivest, "A digital signature scheme secure against adaptive chosenmessage attacks," *SIAM Journal of Computing*, vol. 17, no. 2, pp. 281-308, 1988.

[24] L. G. Guillou, and J. J. Quisquater. "A 'paradoxical' identity-based signature scheme resulting from zero-knowledge," *Proceedings of Crypto' 88*, LNCS 403, pp. 216-231, Springer-Verlag, 1990.

[25] S. Guo, Z. Cao, and R. Lu, "An efficient ID-based multi-proxy multi-signature scheme," *Proceedings of International Multi-Symposiums on Computer and Computational Sciences*, pp. 81-88, IEEE Computer Society, 2006.

[26] L. Harn, "Goup oriented (t, n) digital signature scheme and digital multi-signature," *IEE Proceedings Computers & Digital Techniques*, vol. 141, no. 5, pp. 307-313, 1994.

[27] F. Hess, "An efficient identity based signature schemes based on pairings," *Proceedings of SAC 2002*, LNCS 2595, pp. 310-324, Springer-Verlag, 2002.

[28] J. Herranz, and G. Saez, "Fully distributed proxy signature schemes", 2003. (http://eprint.iacr.org/2003/53/)

[29] J. Herranz, and G. Saez, "Verifiable secret sharing for general access structures, with applications to fully distributed distributed proxy signatures," *Proceedings of Financial Cryptography (FC' 03)*, LNCS 2742, pp. 286-302, Springer-Verlag, 2003.

[30] J. Herranz, and G. Saez, "Revisiting fully distributed proxy signature schemes," *Proceedings of Indocrypt '04*, LNCS 3348, pp. 356-370, Springer-Verlag, 2004.

[31] C. L. Hsu, T. S. Wu, and T. C. Wu, "Improvement of threshold proxy signature scheme," *Applied Mathematics & Computation*, vol. 136, pp. 315-321, 2003.

[32] Z. Huang, and Y. Wang, "Convertible nominative signatures," *Proceedings of Australasian Conference on Information Security and Privacy (ACISP'04)*, LNCS 3108, pp. 348-357, Springer-Verlag, 2004.

[33] S. Hwang, and C. Chen, "Cryptanalysis of nonrepudiable threshold proxy signature schemes with known signers," *Informatica*, vol. 14, no. 2, pp. 205-212, 2003.

[34] M. Hwang, I. Lin, and E. J. Lu, "A secure nonrepudiable threshold proxy signature scheme with known signers," *Informatica*, vol. 11, no. 2, pp. 1-8, 2000.

[35] M. S. Hwang, E. J. L. Lu, and I. C. Lin, "A practical (t, n) threshold proxy signature scheme based on RSA cryptosystem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 6, pp. 1552-1560, 2003.

[36] M. A. Ibrahim, and A. Cerny, "Proxy and threshold one-time signatures," *Proceedings of International Conference Applied Cryptography and Network Security (ACNS' 03)*, LNCS 2846, pp. 123-136, Springer-Verlag, 2003.

[37] A. Ivan, and Y. Dodis, "Proxy cryptography revisited," *Proceedings of Network and Distributed System Security Symposium (NDSS'03)*, The Internet Society, 2003.

[38] S. Kim, S. Park, and D. Won, "Proxy signatures revisited," *Proceedings of Information and Communications Security (ICICS' 97)*, LNCS 1334, Springer-Verlag, pp. 223-232, 1997.

[39] S. Lal, and A. K. Awasthi, "A scheme for obtaining a Warrant message from the digital proxy signatures," 2003. (http://eprint.iacr.org/2003/073/)

[40] J. Lee, J. Cheon, and S. Kim, "An analysis of proxy signatures: Is a secure channel necessary?," *Proceedings of CT-RSA Conference*, LNCS 2612, pp. 68-79, Springer-Verlag, 2003.

[41] N. Y. Lee, T. Hwang, and C. H. Wang, "On Zhang's nonrepudiable proxy signature schemes," *Proceedings of Australasian Conference on Information Security and Privacy (ACISP' 98)*, LNCS 1438, pp. 415-422, Springer-Verlag, 1998.

[42] N. Y. Lee, and M. F. Lee, "The security of a strong proxy signature scheme with proxy signer privacy protection," *Applied Mathematics and Computation*, vol. 161, pp. 807-812, 2005.

[43] B. Lee, and K. Kim, "Strong proxy signatures," *IEICE Transactions Fundamentals*, vol. E-82, no. 1, pp. 1-11, 1999.

[44] B. Lee, H. Kim, and K. Kim, "Strong proxy signature and its applications," *Proceedings of Symposium on Cryptography and Information Security*, pp. 603-608, 2001.

[45] B. Lee, H. Kim, and K. Kim, "Secure mobile agent using strong non-designated proxy signature," *Proceedings of Australasian Conference on Information Security and Privacy (ACISP' 01)*, LNCS 2119, pp. 474-486, Springer-Verlag, 2001.

[46] L. Li, S. Tzeng, and M. Hwang, "Generalization of proxy signature-based on discrete logarithms," *Computers & Security*, vol. 22, no. 3, pp. 245-255, 2003.

[47] C. Y. Lin, T. C. Wu, and F. Zhang, "Proxy signature and proxy multi-signature from bilinear pairings," *Proceedings of International Conference on Informatics, Cybernetics and Systems*, 2003.

[48] R. Lu, Z. Cao, and X. Dong, "Efficient ID-based one-time proxy signature and Its application in E-cheque," *Proceedings of Cryptology and Network Security*, LNCS 4301, pp. 153-167, Springer-Verlag, 2006

[49] E. J. L. Lu, and C. J. Huang, "A time-stamping proxy signature scheme using time-stamping ser-

vice," *International Journal of Network Security*, vol. 2, no. 1, pp. 43-51, 2006.

[50] R. Lu, Z. Cao, X. Dong, and R. Su, "Designated verifier proxy signature scheme from bilinear pairings," *Proceedings of International Multi-Symposiums on Computer and Computational Sciences*, *IEEE Computer Society*, pp. 40-47, 2006.

[51] J. Lv, J. Liu, and X. Wang, "Further cryptanalysis of some proxy signature schemes," 2003. (http://eprint.iacr.org/2003/111/)

[52] T. Malkin, S. Obana, and M. Yung, "The hierarchy of key evolving signatures and a characterization of proxy signatures," *Proceedings of Eurocrypt' 04*, LNCS 3027, Springer-Verlag, pp. 306-322, 2004.

[53] M. Mambo, K. Usuda, and E. Okamoto, "Proxy Signatures: Delegation of the Power to Sign Messages," *IEICE Transactions Fundamentals*, vol. E79-A, no. 9, pp. 1338-1353, 1996.

[54] A. Menezes, T. Okamoto, and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in finite field," *IEEE Transactions on Information Theory*, vol. 39, no. 5, pp. 1639-1646, 1993.

[55] A. Menezes, P. C. V. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[56] T. Okamoto, A. Inomata, and E. Okamoto, "A proposal of short proxy signature using pairing," *International Conference on Information Technology: Coding and Computing (ITCC'05)*, pp. 631-635, IEEE Computer Society, 2005.

[57] T. Okamoto, M. Tada, and E. Okamoto, "Extended proxy signaures for smart card" *Proceedings of Information Security Workshop'99*, LNCS 1729, pp. 247-258, Springer-Verlag, 1999.

[58] H. U. Park, and I. Y. Lee, "A digital nominative proxy signature scheme for mobile communications," *Proceedings of Information and Communications Security (ICICS' 01)*, LNCS 2229, pp. 451-455, Springer-Verlag, 2001.

[59] T. Pedersen, "Distributed provers with applications to undeniable signatures," *Proceedings of Eurocrypt' 91*, LNCS 547, Springer-Verlag, pp. 221-238, 1991.

[60] H. Petersen, and P. Horster, "Self-certified keys-concepts and applications," *Proceedings of Conference on Communivations and Multimedia Security*, pp. 102-116, 1997.

[61] H. Qian, and Z. Cao, "A novel ID-based partial delegation with Warrant proxy signature scheme," *Proceedings of ISPA Workshops*, LNCS 3759, pp. 323-331, Springer-Verlag, 2005.

[62] M. O. Rabin, "Digitalized signatures," *Foundations of Secure Communication*, pp. 155-168, Academic Press, 1978.

[63] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[64] A. Romao, and M. M. da Silva, "Secure mobile sgent digital signatures with proxy certificates," *Proceedings of E-Commerce Agents*, LNAI 2033, pp. 206-220, 2001.

[65] Z. Shao, "Proxy signature schemes based on factoring," *Information Processing Letters*, vol. 85, pp. 137-143, 2003.

[66] Z. Shao, "Improvement of threshold proxy signature scheme," *Computer Standards & Interfaces*, vol. 27, pp. 53-59, 2004.

[67] K. Shum, and V. K. Wei, "A strong proxy signature scheme with proxy signer privacy protection," *Proceedings of IEEE International Workshop on Enabling Technologies: Infrastucture for Collaborative Enterprises (WETICE' 02)*, 2002.

[68] C. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptography*, vol. 4, no. 3, pp. 161-174, 1991.

[69] H. M. Sun, "An efficient nonrepudiable threshold proxy signature scheme with known signers," *Computer Communications*, vol. 22, no. 8, pp. 717-722, 1999.

[70] H. M. Sun, "On proxy multi-signature scheme," *Proceedings of the International Computer Symposium*, pp. 65-72, 2000.

[71] H. M. Sun, "Design of time-stamped proxy signatures with traceable receivers," *IEE Proceedings of Computers & Digital Techniques*, vol. 147, no. 6, pp. 462-466, 2000.

[72] H. M. Sun, and B. T. Hsieh, "On the security of some proxy signature schemes," 2003. (http://eprint.iacr.org/2003/068/)

[73] H. M. Sun, N. Y. Lee, and T. Hwang, "Threshold proxy signatures," *IEE Proceedings of Computers & Digital Techniques*, vol. 146, no. 5, pp. 259-263, 1999.

[74] Z. Tan, and Z. Liu, "Provably secure delegation-by-certification proxy signature schemes," 2004. (http://eprint.iacr.org/2004/148/)

[75] Z. Tan, and Z. Liu, "On the security of some nonrepudiable threshold proxy signature schemes with known signers," 2004. (http://eprint.iacr.org/2004/234/)

[76] C. S. Tsai, S. F. Tzeng, and M. S. Hwang, "Improved non-repudiable threshold proxy signature scheme with known Signers," *Informatica*, vol. 14, no. 3, pp. 1-10, 2003.

[77] K. Viswanathan, C. Boyd, and E. Dwason, "Signature scheme for sontrolled environments," *Proceedings of Information and Communications Security (ICICS' 99)*, LNCS 1726, pp. 119-134, Springer-Verlag, 1999.

[78] G. Wang, "Designated-verifier proxy signatures for e-commerce,", *Proceedings of International Conference on Multimedia and Expo (ICME' 04)*, pp. 1731-1734, 2004.

[79] G. Wang, "Designated-verifier proxy signature schemes," *Proceedings of Security and Privacy in the Age of Ubiquitous Computing*, IFIP series 181, pp. 409-423, Springer-Verlag, 2005.

[80] G. Wang, F. Bao, J. Zhou, and R. H. Deng, "Security analysis of some proxy signatures," 2003. (http://eprint.iacr.org/2003/196/)

[81] G. Wang, F. Bao, J. Zhou, and R. H. Deng, "Comments on a practical (t, n) threshold proxy signature scheme based on the RSA cryptosystem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 10, pp. 1309-1311, 2004.

[82] J. Xu, Z. Zhang, and D. Feng, "ID-based proxy signature using bilinear pairings," 2004. (http://eprint.iacr.org/2004/206/)

[83] L. Yi, G. Bai, and G. Xiao, "Proxy multi-signature scheme: A new type of proxy signature scheme,", *Electronics Letters*, vol. 36, no. 6, pp. 527-528, 2000.

[84] K. Zhang, "Nonrepudiable proxy signature schemes," 2008. (http://citeseer.nj.nec.com/360090.html/)

[85] K. Zhang, "Threshold proxy signature schemes," *Proceedings of Information Security Workshop*, LNCS 1396, pp. 191-197, Springer-Verlag, 1997.

[86] F. Zhang, and K. Kim, "Efficient ID-based blind signature and proxy signature from bilinear pairings," *Proceedings of Australasian Conference on Information Security and Privacy (ACISP'2003)*, LNCS 2727, pp. 312-323, Springer-Verlag, 2003.

[87] F. Zhang, R. S. Naini, and C. Y. Lin, "New proxy signature, proxy blind signature and proxy ring signature schemes from bilinear pairing," 2003. (http://eprint.iacr.org/2003/104/)

[88] F. Zhang, R. Safavi-Naini, and W. Susilo, "An efficient signature from bilinear pairings and its applications," *Proceedings of Public Key Crptography (PKC'04)*, LNCS 2947, pp. 277-290, Springer-Verlag, 2004.

[89] Y. Zhou, Z. Cao, and Z. Chai, "An efficient proxy-protected signature scheme based on factoring," *Proceedings of ISPA Workshops*, LNCS 3759, pp. 332-341, Springer-Verlag, 2005.

[90] Y. Zhou, Z. Cao, and Z. Chai, "Constructing secure proxy cryptosystem," *Proceedings of CISC*, LNCS 3822, pp. 150-161, Springer-Verlag, 2005.

Manik Lal Das received his Ph.D. degree from Indian Institute of Technology, Bombay, India in 2006. He is an Assistant Professor in Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar, India. He has published over 40 research articles in refereed Journals/Conferences. He is a member of the IEEE, Cryptology Research Society of India and Indian Society for Technical Education. His research interests include Cryptography and Information Security.

Ashutosh Saxena received his M.Sc. (1990), M. Tech. (1992) and Ph.D. in Computer Science (1999). Presently, he is working as Senior Research Associate in the SET Labs at Infosys Technologies Ltd., Hyderabad. He is on the Editorial Committees of various International Journals and Conferences, and is a Life Member of Cryptology Research Society of India and Computer Society of India and Member of IEEE Computer Society. He has to his credit more than 70 research papers published in National/ International Journals and Conferences. His main research interest is in the areas of Authentication Technologies, Smart Card, Key Management and Security Issues and Payment Systems in Banking.

Deepak B Phatak received his Ph.D. degree from Indian Institute of Technology, Bombay, India. He is a Professor in Computer Science & Engineering, Indian Institute of Technology, Bombay, India. His research interests include Data Bases, System performance evaluation, Smart Cards and Information Systems.