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

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