10-2013

K-time proxy signature: Formal definition and efficient construction

Weiwei LIU

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Yi MU

Jiannan WEI

Citation

LIU, Weiwei; YANG, Guomin; MU, Yi; and WEI, Jiannan. K-time proxy signature: Formal definition and efficient construction. (2013). *Proceedings of the 7th International Conference, Melaka, Malaysia, 2013 October 23-25*. 154-164.

Available at: https://ink.library.smu.edu.sg/sis_research/7378

k-time Proxy Signature: Formal Definition and Efficient Construction

Weiwei Liu, Guomin Yang, Yi Mu, and Jiannan Wei

School of Computer Science and Software Engineering,
University of Wollongong, Wollongong, NSW 2522, Australia
{wl265,jw903}@uowmail.edu.au, {gyang,ymu}@uow.edu.au

Abstract. Proxy signature, which allows an original signer to delegate his/her signing right to another party (or proxy signer), is very useful in many applications. Conventional proxy signature only allows the original signer to specify in the warrant the validity time period of the delegation but not the number of proxy signatures the proxy signer can generate. To address this problem, in this paper, we provide a formal treatment for *k-time proxy signature*. Such a scheme allows a designated proxy signer to produce only a fixed number of proxy signatures on behalf of the original signer. We provide the formal definitions and adversary models for k-time proxy signature, and propose an efficient construction which is *provably secure* against different types of adversaries.

Keywords: proxy signature, restricted delegation, secret sharing.

1 Introduction

Proxy signature is a special type of digital signature, and is very useful in many real-world applications. In a proxy signature scheme, an original signer (or delegator) can delegate his/her signing right to a proxy signer. Thereafter, the proxy signer can sign documents on behalf of the original signer.

The first proxy signature scheme was proposed by Mambo, Usuda and Okamoto in 1996 [14]. In their work they classified proxy signatures into three main categories, namely full delegation, partial delegation, and delegation by warrant. Partial delegation proxy signature schemes can be further divided into proxy-protected and proxy-unprotected schemes according to whether a verifier can decide the proxy signature is generated by a proxy signer or the original signer. Shortly after that, Kim et al. [10] proposed a new type of proxy signature combining partial delegation and warrant. They further showed that such a combination can provide a higher level of security. Since then many proxy signature schemes based on partial delegation and warrant have been proposed (e.g., [12,24,20,23,25]).

Many extensions on proxy signature have also been proposed according to different application needs, such as threshold proxy signature [28,26,13], blind proxy signature [27,4,2], one-time proxy signature [15,21], ring proxy signature

W. Susilo and R. Reyhanitabar (Eds.): ProvSec 2013, LNCS 8209, pp. 154–164, 2013.

[22,1,7], and so on. Threshold proxy signature, also known as multi-proxy signature, enables an original signer to delegate his signing right to multiple proxy signers. The proxy signers need to work together in order to produce a valid proxy signature on behalf of the original signer. One-time proxy signature puts strict restrictions on the signing capability of a proxy signer, who is only allowed to generate one valid proxy signature on behalf of the original signer. Blind proxy signature allows a user to obtain a valid signature on a message in a way that the proxy signer learns neither the message nor the resulting signature, and ring proxy signature allows a proxy signer to hide his/her identity among a group of possible signers.

Proxy signature and its extended variants have been found very useful in many practical applications, such as distributed systems [16], grid computing [6], and mobile agent applications [12]. However, one of the key issues in proxy signature is to ensure that a proxy signer will not misuse the signing right obtained from an original signer. In the seminal work by Mambo et al. [14], a validity period is specified in a warrant in order to restrict the signing capability of a proxy signer. This approach has been used in almost all the following works on proxy signature. However, if the proxy signer is malicious, even in a very short time, the malicious proxy signer can still produce as many proxy signatures as he/she wishes. To address this problem, in this paper, we provide a formal and comprehensive treatment for k-time proxy signature where the proxy signer can only generate a fixed number of proxy signatures on behalf of the original signer.

There have been a number of works (e.g., [3,9,18,11]) on restricting the signing capability of a signer in normal digital signature schemes. In [9], Hwang et al. proposed a multiple-time digital signature scheme, which gives an upper bound on the number of signatures a signer can produce. Shortly after that, Pieprzyk et al. [18] proposed a more general multiple-time signature scheme based on one-way functions and cover-free families. Kim et al. [11] then extended multiple-time signature to a new primitive named metered signature, which allows a signer to produce a fixed number of signatures in a designated time period.

However, a formal and complete treatment for multi-time (or k-time) proxy signature is still missing. In [15], Mehta and Harn proposed a one-time proxy signature scheme, which is less useful than a more general k-time proxy signature scheme. There is a multi-time proxy signature scheme presented in [5], however, no formal security model or proof has been provided. In [8], Hong and Chen presented a multiple-time proxy signature scheme based on a binary hash tree. However, their security analysis is incomplete since it does not cover all the possible attacks against a multiple-time proxy signature scheme.

In this paper, we provide a formal and complete treatment for multi-time (or k-time) proxy signature schemes. We first provide a formal security model for such schemes. In our model, we will consider three types of adversaries, namely outsiders, proxy signer, and original signer. Our model aims to capture the exact security goal of a k-time proxy signature scheme, that is only a proxy signer, who has been delegated the signing right from an original signer, can produce *at most* k valid proxy signatures. We then propose a new k-time proxy signature scheme based on the Schnorr signature scheme and verifiable secret sharing. In our scheme, the original signer can specify in the warrant the number of proxy signatures a proxy signer can produce. If the proxy signer produces more than predetermined number of proxy signatures, his/her private key can be computed by the public. That means the original signer does not need to monitor the behavior of the proxy signer. It is worth noting that such a feature is not supported in Hong and Chen's scheme [8]. In their scheme, the proxy signer's private key can only be computed by the original signer rather than by any third party verifier when the proxy signer misbehaves.

Paper Outline. The rest of the paper is organized as follows. We introduce the definition of k-time proxy signature in Section 2. A formal security model for k-time proxy signature is presented in Section 3. We then give our new proxy signature scheme in Section 4 and prove its security in Section 5. The paper is concluded in Section 6.

2 k-time Proxy Signature

A k-time (or multi-time) proxy signature scheme consists of a tuple of algorithms $(\mathcal{ST}, \mathcal{KG}, \mathcal{DSK}\ \mathcal{PKG}, \mathcal{PS}, \mathcal{PV}, \mathcal{R})$:

- Setup–($\mathcal{ST}$): This algorithm takes 1^κ as input where κ is a security parameter and returns the public parameters $params$.
- KeyGen–($\mathcal{KG}$): The Key Generation algorithm takes $params$ as input and outputs a user key pair (pk, sk).
- DskGen–($\mathcal{DKG}$): This algorithm takes (sk_o, pk_o, pk_p, m_w) as input and outputs a delegation key dsk. Here m_w denotes a warrant which specifies the predetermined number of proxy signatures that can be generated by the proxy signer.
- PskGen–($\mathcal{PKG}$): This algorithm takes dsk and sk_p as input and outputs a proxy signing key psk.
- ProSig–($\mathcal{PS}$): The proxy signing algorithm takes the proxy signing key psk and a message m in the message space $\mathbb{M}$ as input, and outputs a proxy signature σ.
- ProVer–($\mathcal{PV}$): The proxy signature verification algorithm takes the public keys pk_o and pk_p, a warrant m_w, a message m, and a proxy signature σ as input, and outputs either 1 or 0.
- Reveal–($\mathcal{R}$): Given pk_o, pk_p, a warrant m_w, and $k+1$ different message and proxy signature pairs, where k is the number specified in the warrant m_w, this algorithm either outputs a private key sk_p of the proxy signer or a special symbol '$\perp$'.

Correctness. We require that for any message space $\mathbb{M} \subseteq \{0,1\}^*$ and any security parameter $\kappa \in \mathbb{N}$, if $params \leftarrow \mathcal{ST}(1^\kappa)$, $(sk_o, pk_o) \leftarrow \mathcal{KG}(params)$, $(sk_p, pk_p) \leftarrow \mathcal{KG}(params)$, $dsk \leftarrow \mathcal{DKG}(sk_o, pk_o, pk_p, m_w)$, $psk \leftarrow \mathcal{PKG}(dsk, sk_p)$, then

$$\mathcal{PV}(pk_o, pk_p, m_w, m, \mathcal{PS}(psk, m)) = 1.$$

3 Security Model

In a k-time proxy signature scheme, the security consideration is different from that for the traditional proxy signature [25] or k-time signature [9]. According to the definition, the security of a k-time proxy signature should be defined in three aspects, which are summarized below.

1. Type I: the Type I attacker $\mathcal{A}_I$ (an outsider) possesses the public keys of the original signer and the proxy signer, and tries to forge a proxy signature.
2. Type II: the Type II attacker $\mathcal{A}_{II}$ (proxy signer) possesses the public keys of the original signer and the proxy signer. In addition, he also possesses the private key sk_p. We can further divide $\mathcal{A}_{II}$ into $\mathcal{A}_{II1}$ and $\mathcal{A}_{II2}$. $\mathcal{A}_{II1}$ tries to forge a valid proxy signature without obtaining delegation from the original signer, and $\mathcal{A}_{II2}$ has a valid delegation from the original signer and tries to produce more than predetermined number of proxy signatures.
3. Type III: the Type III attacker $\mathcal{A}_{III}$ (the original signer) possesses the public keys of the original signer and the proxy signer. In addition, he has the private key sk_o of the original signer. $\mathcal{A}_{III}$ tries to forge a valid proxy signature without knowing the private key sk_p of the proxy signer.

It is obvious that if a k-time proxy signature scheme is secure against $\mathcal{A}_{II}$ and $\mathcal{A}_{III}$, it is also secure against $\mathcal{A}_I$. So we will only focus on the adversarial models with regards to $\mathcal{A}_{II}$ and $\mathcal{A}_{III}$ in the rest of this paper.

Before we formally define each adversarial model, we first introduce two types of queries that may appear in the models:

- Delegation query: $\mathcal{A}$ can query the delegation oracle $\mathcal{O}_{DKG}(sk_o, pk_o, pk_p, \cdot)$ with any warrant m_w. The corresponding delegation key dsk is then generated and returned to $\mathcal{A}$.
- Proxy signing query: $\mathcal{A}$ can query the proxy signing oracle $\mathcal{O}_{PS}(psk, \cdot)$ with any message m of his choice. A valid proxy signature on m is generated and returned to $\mathcal{A}$.

3.1 Type II1 Adversary

We define the adversarial game between a Type II1 adversary $\mathcal{A}_{II1}$ and an simulator $\mathcal{S}$ as follows:

- **Setup:** The Simulator $\mathcal{S}$ runs $\mathcal{ST}$ to generate public parameters $params$.
- **KeyGen** The Simulator $\mathcal{S}$ runs $\mathcal{KG}$ to generate the key pairs of the original signer (sk_o, pk_o) and a proxy signer (sk_p, pk_p). $\mathcal{S}$ sends pk_o, pk_p and sk_p to the adversary $\mathcal{A}_{II1}$.
- **Delegation queries:** $\mathcal{A}_{II1}$ chooses any warrant m_w of his/her choice and queries the delegation oracle $\mathcal{O}_{DKG}$. $\mathcal{S}$ generates the delegation key $dsk \leftarrow \mathcal{DKG}(sk_o, pk_o, pk_p, m_w)$ and returns dsk to $\mathcal{A}_{II1}$.

- **Proxy signing queries:** $\mathcal{A}_{II1}$ chooses a warrant m_w and a message m, and queries the proxy signing oracle $\mathcal{O}_{PS}$. If m_w has appeared in a Delegation Query, a special symbol '$\perp$' is returned to $\mathcal{A}$. Otherwise, $\mathcal{S}$ generates $dsk \leftarrow \mathcal{DKG}(sk_o, pk_o, pk_p, m_w)$, $psk \leftarrow \mathcal{PKG}(dsk, sk_p)$, $\sigma \leftarrow \mathcal{PS}(psk, m)$, and returns σ to $\mathcal{A}_{II1}$.
- Finally, $\mathcal{A}_{II1}$ outputs (m_w^*, m^*, σ^*). We say $\mathcal{A}_{II1}$ wins the game if
  - $\mathcal{PV}(pk_o, pk_p, m_w^*, m^*, \sigma^*) = 1$;
  - $\mathcal{A}_{II1}$ did not make a query to $\mathcal{O}_{DKG}$ on m_w^*;
  - $\mathcal{A}_{II1}$ did not make a query to $\mathcal{O}_{PS}$ on (m_w^*, m^*).

Define the advantage of a Type II1 adversary as

$$Adv_{\mathcal{A}_{II1}}^{cwcma}(\kappa) = \Pr[A_{II1} \text{ Wins the game}].$$

Definition 1. *We say a k-time proxy signature scheme is secure against the Type $II1$ chosen warrant and chosen message attacks if for any probabilistic polynomial time $\mathcal{A}_{II1}$, $Adv_{\mathcal{A}_{II1}}^{cwcma}(\kappa)$ is negligible in κ.*

3.2 Type II2 Adversary

We define the adversarial game between a Type II2 adversary $\mathcal{A}_{II2}$ and an simulator $\mathcal{S}$ as follows:

- **Setup:** The Simulator $\mathcal{S}$ runs $\mathcal{ST}$ to generate public parameters *params*.
- **KeyGen** The Simulator $\mathcal{S}$ runs $\mathcal{KG}$ to generate the key pairs of an original signer (sk_o, pk_o) and a proxy signer (sk_p, pk_p). $\mathcal{S}$ sends pk_o, pk_p and sk_p to the adversary $\mathcal{A}_{II2}$.
- **Delegation queries:** $\mathcal{A}_{II2}$ chooses any warrant m_w of his/her choice and queries the delegation oracle $\mathcal{O}_{DKG}$. $\mathcal{S}$ generates the delegation key $dsk \leftarrow \mathcal{DKG}(sk_o, pk_o, pk_p, m_w)$ and returns dsk to $\mathcal{A}_{II2}$.
- Finally, $\mathcal{A}_{II2}$ outputs a warrant m_w which contains a predetermined number k, and $k+1$ message-signature pairs (m_i, σ_i) $(1 \leq i \leq k+1)$ where $m_i \neq m_j$ for $i \neq j$. We say $\mathcal{A}_{II2}$ wins the game if
  - $\mathcal{PV}(pk_o, pk_p, m_w, m_i, \sigma_i) = 1$ for all $i \in [1, k+1]$;
  - $\mathcal{R}(pk_o, pk_p, m_w, (m_1, \sigma_1), \cdots, (m_{k+1}, \sigma_{k+1})) = \perp$.

Define the advantage of a Type II2 adversary as

$$Adv_{\mathcal{A}_{II2}}^{cwa}(\kappa) = \Pr[A_{II2} \text{ Wins the game}].$$

Definition 2. *We say a k-time proxy signature scheme is secure against the Type $II2$ chosen warrant attacks if for any probabilistic polynomial time $\mathcal{A}_{II2}$, $Adv_{\mathcal{A}_{II2}}^{cwa}(\kappa)$ is negligible in κ.*

3.3 Type III Adversary

The adversarial game between a Type III adversary $\mathcal{A}_{III}$ and an simulator $\mathcal{S}$ is defined as follows:

- **Setup:** The Simulator $\mathcal{S}$ runs $\mathcal{S}$ to generate public parameters *params* and gives *params* to the adversary.
- **KeyGen** The Simulator $\mathcal{S}$ runs $\mathcal{KG}$ to generate the key pairs of the original signer (sk_o, pk_o) and a proxy signer (sk_p, pk_p). $\mathcal{S}$ sends sk_o, pk_o and pk_p to the adversary $\mathcal{A}_{III}$.
- **Proxy signing queries:** $\mathcal{A}_{III}$ queries the proxy signing oracle $\mathcal{O}_{PS}$ by providing a warrant m_w generated according to the scheme, a valid delegation key dsk for m_w, and a message m. $\mathcal{S}$ generates $psk \leftarrow \mathcal{PKG}(dsk, sk_p)$, $\sigma \leftarrow \mathcal{PS}(psk, m)$, and returns σ to $\mathcal{A}_{III}$.
- Finally, $\mathcal{A}_{III}$ outputs (m_w^*, m^*, σ^*). We say $\mathcal{A}_{III}$ wins the game if
  - $\mathcal{PV}(pk_o, pk_p, m_w^*, m^*, \sigma^*) = 1$;
  - For any warrant m_w with a predetermined number k, $\mathcal{A}_{III}$ makes at most k proxy signing queries;
  - $\mathcal{A}_{III}$ did not make a query to $\mathcal{O}_{PS}$ on (m_w^*, m^*).

Define the advantage of a Type III adversary as

$$Adv_{\mathcal{A}_{III}}^{cma}(\kappa) = \Pr[A_{III} \text{ Wins the game}].$$

Definition 3. *We say a k-time proxy signature scheme is secure against the Type III chosen message attacks if for any probabilistic polynomial time $\mathcal{A}_{III}$, $Adv_{\mathcal{A}_{III}}^{cma}(\kappa)$ is negligible in κ.*

4 A New *k*-time Proxy Signature Scheme

In this section, we present a new k-time proxy signature scheme based on the Discrete Logarithm Problem and secret sharing.

Discrete Logarithm Problem (DLP): Let G denote a group of prime order q, and g a generator of G. Given a random element $y \in G$, compute $x \in \mathbb{Z}_q$ such that $y = g^x$.

Our k-time proxy signature scheme works as follows:

1. $\mathcal{ST}$: given a security parameter $\kappa \in \mathbb{N}$, generate the parameters $params = (G, g, q)$ such that $|q| = \kappa$ and a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$.
2. $\mathcal{KG}$: randomly choose $x \in \mathbb{Z}_q$ and compute $y = g^x$. Output $(sk, pk) = (x, y)$.
3. $\mathcal{DKG}$: given a warrant $m_w = (k, B = \{b_1, b_2, \cdots, b_k\})$[1], where k is a number selected by the original signer and $b_i = g^{a_i}$ $(1 \leq i \leq k)$ are generated by the proxy signer and sent to the original singer via a secure channel, the original signer first chooses a random number $k_o \in \mathbb{Z}_q$, and then computes $K_o = g^{k_o}$, $\sigma_o = sk_o \cdot h(m_w \| K_o) + k_o \bmod q$. The original signer then sets $dsk = (K_o, \sigma_o)$ as the delegation key for m_w.

[1] It is worth noting that we can put additional information, such as the validity time period and the type of message the proxy signer is allowed to sign, in the warrant.

4. $\mathcal{PKG}$: given a delegation key $dsk = (K_o, \sigma_o)$ for a warrant m_w, the proxy signer computes $S_p = \sigma_o + sk_p \bmod q$ and outputs the proxy signing key $psk = (K_o, S_p, sk_p)$.
5. $\mathcal{PS}$: given a message m to be signed, and a proxy signing key $psk = (K_o, S_p, sk_p)$, the proxy signer chooses a random number $k_p \in \mathbb{Z}_q$, and computes $K_p = g^{k_p}$ and $\sigma_p = S_p \cdot h(h(m_w \| K_o)\|m\|K_p) + k_p \bmod q$. The proxy signer also computes $f(\omega) = sk_p + a_1\omega + a_2\omega^2 + ... + a_k\omega^k \bmod q$ where $\omega = h(m_w, m, \sigma_p)$. The proxy signature is $\sigma = (K_o, K_p, \sigma_p, f(\omega))$.
6. $\mathcal{PV}$: given public keys pk_o and pk_p, a warrant $m_w = (k, B = \{b_1, b_2, \cdots, b_k\})$, a message m and a proxy signature $\sigma = (K_o, K_p, \sigma_p, f(\omega))$, the verifier checks if the following equation holds
   - $g^{\sigma_p} = K_p \cdot (pk_p \cdot K_o \cdot pk_o^{h(m_w\|K_o)})^{h(h(m_w\|K_o)\|m\|K_p)}$;
   - $g^{f(\omega)} = pk_p \cdot b_1^{\omega} \cdot b_2^{\omega^2} \cdots b_k^{\omega^k}$.

   If both equations hold, output 1; otherwise, output 0.
7. $\mathcal{R}$: given $pk_o, pk_p, m_w = (k, B = \{b_1, b_2, \cdots, b_k\})$, and $k+1$ message signature pairs (m_i, σ_i), solve the following equations

$$\begin{aligned} f(\omega_1) &= sk_p + a_1\omega_1 + a_2\omega_1^2 + ... + a_k\omega_1^k \\ f(\omega_2) &= sk_p + a_1\omega_2 + a_2\omega_2^2 + ... + a_k\omega_2^k \\ &\cdots \\ f(\omega_{k+1}) &= sk_p + a_1\omega_{k+1} + a_2\omega_{k+1}^2 + ... + a_k\omega_{k+1}^k \end{aligned}$$

   for variables $(sk_p, a_1, \cdots, a_k)$. If a solution is found, output sk_p, otherwise, output '$\perp$'.

The correctness of the scheme can be verified as follows

$$\begin{aligned} g^{\sigma_p} &= g^{S_p \cdot h(h(m_w\|K_o)\|m\|K_p) + k_p} \\ &= (g^{\sigma_o + sk_p})^{h(h(m_w\|K_o)\|m\|K_p)} \cdot g^{k_p} \\ &= (g^{sk_o \cdot h(m_w\|K_o) + k_o} \cdot g^{sk_p})^{h(h(m_w\|K_o)\|m\|K_p)} \cdot K_p \\ &= (pk_o^{h(m_w\|K_o)} \cdot K_o \cdot pk_p)^{h(h(m_w\|K_o)\|m\|K_p)} \cdot K_p \\ g^{f(\omega)} &= g^{sk_p + a_1\omega + a_2\omega^2 + ... + a_k\omega^k} \\ &= pk_p \cdot (g^{a_1})^{\omega} \cdot (g^{a_2})^{\omega^2} \cdots (g^{a_k})^{\omega^k} \\ &= pk_p \cdot b_1^{\omega} \cdot b_2^{\omega^2} \cdots b_k^{\omega^k} \end{aligned}$$

5 Security Analysis

In this section we analyse the security of the above k-time proxy signature scheme against $\mathcal{A}_{II}$ and $\mathcal{A}_{III}$ adversaries.

Theorem 1. *The proposed k-time proxy signature scheme is secure against the Type $II1$ chosen warrant and chosen message attacks if the Discrete Logarithm Problem is hard.*

Proof. The proof is by contradiction. Given an adversary $\mathcal{A}_{II1}$ that can win the Type $II1$ game, we construct another algorithm $\mathcal{B}$ that can solve the DLP.

Given $(g, y^* = g^{x^*})$ for some unknown $x^* \in \mathbb{Z}_q$, $\mathcal{B}$ simulates the Type II1 game for $\mathcal{A}_{II1}$ as follows. $\mathcal{B}$ sets the original signer's public key as $pk_o = y^*$ and maintains a H-table to record all the hash queries and the corresponding answers.

Hash Queries: For each hash query with an input message msg, $\mathcal{B}$ first checks the H-table:

- If there exists an item (msg, h) in the H-table,where msg refers to the messages queried before, $\mathcal{B}$ returns h as the answer to $\mathcal{A}_{II1}$.
- Otherwise, $\mathcal{B}$ chooses a random $h \in Z_q$, sends h to $\mathcal{A}_{II1}$ as the answer for the hash query, and adds (msg, h) into the H-table.

Delegation Queries: When $\mathcal{A}_{II1}$ makes a delegation query on a warrant $m_w = (k, B = (b_1, b_2, \cdots, b_k))$, $\mathcal{B}$ answers the query as follows.

- Choose randomly $h_o, \sigma_o \in Z_q$, compute $K_o = g^{\sigma_o}/pk_o^{h_o}$, and set $h(m_w \| K_o) = h_o$ by adding $(m_w \| K_o, h_o)$ into the H-table.
- Return (K_o, σ_o) as the delegation key to $\mathcal{A}_{II1}$.

Proxy Signing Queries: When $\mathcal{A}_{II1}$ makes a proxy signing query on a warrant $m_w = (k, B = (b_1, b_2, \cdots, b_k))$, and a message m, $\mathcal{B}$ responds the query as follows:

- Generate a delegation key $dsk = (K_o, \sigma_o)$ for the warrant m_w by applying the same approach as described in answering delegation queries.
- Use the derived dsk and sk_p to produce the proxy signing key psk by running the $\mathcal{PKG}$ algorithm, and then use psk to generate the proxy signature for message m by running the $\mathcal{PS}$ algorithm.

Assume $\mathcal{A}_{II1}$ can forge a valid proxy signature $\sigma^* = (K_o^*, K_p^*, \sigma_p^*, f(\omega^*))$ for a warrant m_w^* and a message m^* such that

$$g^{\sigma_p^*} = K_p^* \cdot (pk_p \cdot K_o^* \cdot pk_o^{h(m_w^* \| K_o^*)})^{h(h(m_w^* \| K_o^*) \| m^* \| K_p^*)}.$$

Then according to the Forking Lemma [19], by rewinding the adversary and providing a new hash value for $h(m_w^* \| K_o^*) \| m^* \| K_p^*$, $\mathcal{B}$ can obtain $S_p^* = \sigma_o^* + sk_p \bmod q$ and $\sigma_o^* = S_p^* - sk_p \bmod q$ which satisfies

$$g^{\sigma_o^*} = K_o^* \cdot pk_o^{h^*}$$

where $h^* = h(m_w^* \| K_o^*)$.

After that, $\mathcal{B}$ repeats the above simulation for $\mathcal{A}_{II1}$ except that a new value $\hat{h}^*$ is chosen as the hash value for $m_w^* \| K_o^*$. Again, due to the Forking Lemma, $\mathcal{B}$ can obtain a new $\hat{\sigma}_o^*$ which satisfies

$$g^{\hat{\sigma}_o^*} = K_o^* \cdot pk_o^{\hat{h}^*}.$$

$\mathcal{B}$ can then compute $x^* = sk_o = (\sigma_o^* - \hat{\sigma}_o^*)/(h^* - \hat{h}^*)$ and solve the Discrete Logarithm Problem. This completes the proof for Theorem 1.

Theorem 2. *The proposed k-time proxy signature scheme is secure against the Type $II2$ chosen warrant attacks.*

Proof. According to our scheme, if a signature $\sigma = (K_o, K_p, \sigma_p, f(\omega))$ is valid with regards to a warrant $m_w = (k, B = (b_1, b_2, \cdots, b_k))$ and message m , then

$$g^{f(\omega)} = pk_p \cdot {b_1}^{\omega} \cdot {b_2}^{\omega^2} \cdots {b_k}^{\omega^k}.$$

Suppose an adversary $\mathcal{A}_{II2}$ have produced $k+1$ proxy signatures with regards to a warrant m_w and different messages $\{m_1, m_2, \cdots, m_{k+1}\}$, then we have

$$\begin{cases} f(\omega_1) = sk_p + a_1\omega_1 + a_2\omega_1^2 + ... + a_k\omega_1^k \\ f(\omega_2) = sk_p + a_1\omega_2 + a_2\omega_2^2 + ... + a_k\omega_2^k \\ ... \\ f(\omega_{k+1}) = sk_p + a_1\omega_{k+1} + a_2\omega_{k+1}^2 + ... + a_k\omega_{k+1}^k \end{cases}$$

where $\omega_i = h(m_w, m_i, \sigma_{p_i})$ for $1 \leq i \leq k+1$. Since the hash function is modelled as a random oracle, each ω_i is a random element in $\mathbb{Z}_q$. Therefore, with overwhelming probability, the reveal algorithm $\mathcal{R}$ can recover the unique solution $(sk_p, a_1, a_2, \cdots, a_k)$ that satisfies the above equations.

Theorem 3. *The proposed k-time proxy signature scheme is secure against the Type III chosen message attacks if the Discrete Logarithm Problem is hard.*

Proof. The proof is similar to the proof for Theorem 1, that is, if there exists an adversary $\mathcal{A}_{III}$ which can win the Type III game, we can construct another algorithm $\mathcal{B}$ which can solve the Discrete Logarithm Problem.

Given $(g, y^* = g^{x^*})$ where $x^* \in \mathbb{Z}_q$ is randomly chosen from $\mathbb{Z}_q$, $\mathcal{B}$ simulates the Type III game for $\mathcal{A}_{III}$ as follows. $\mathcal{B}$ generates sk_o, pk_o and sets the proxy signer's public key as $pk_p = y^*$. $\mathcal{B}$ answers hash queries by maintaining a H-table as in the proof of Theorem 1.

When a new warrant m_w with a predetermined number k is to be created, $\mathcal{B}$ generates the values of $B = (b_1, b_2, \cdots, b_k)$ as follows. $\mathcal{B}$ randomly chooses $\omega_i, s_i \in \mathbb{Z}_q$ for $1 \leq i \leq k$. Then based on the result in [17], $\mathcal{B}$ can calculate $b_i (1 \leq i \leq k) \in G$ that satisfies $g^{s_i} = y^* \cdot \prod_{j=1}^{k} b_j^{\omega_i^j}$ for all $1 \leq i \leq k$. $\mathcal{B}$ saves the values of $\{\omega_i, s_i\}_{1 \leq i \leq k}$ with regards to m_w for later use.

Proxy Signing Queries: To answer the ℓ-th $(1 \leq \ell \leq k)$ proxy signing query on a warrant m_w, $\mathcal{B}$ first finds out the values of (ω_ℓ, s_ℓ) that have been computed when generating the warrant m_w. $\mathcal{B}$ then computes the proxy signature as follows:

- Randomly choose $\sigma_p, \tau \in \mathbb{Z}_q$;
- Compute $K_p = g^{\sigma_p} / (pk_p \cdot K_o \cdot pk_o^{h(m_w \| K_o)})^{\tau}$;
- Set $h(h(m_w \| K_o) \| m \| K_p) = \tau$;
- Set $h(m_w \| m \| \sigma_p) = \omega_\ell$;
- Return $\sigma = (K_o, K_p, \sigma_p, s_\ell)$.

It is easy to verify that σ can successfully pass the signature verification.

Suppose $\mathcal{A}_{III}$ outputs a forgery $(m_w^*, m^*, \sigma^* = (K_o^*, K_p^*, \sigma_p^*, s^*))$ which satisfies

$$g^{\sigma_p^*} = K_p^* \cdot (y^* \cdot K_o^* \cdot pk_o^{h(m_w^*\|K_o^*)})^{h(h(m_w^*\|K_o^*)\|m^*\|K_p^*)}$$

where $dsk^* = (K_o^*, \sigma_o^*)$ is the delegation key provided by $\mathcal{A}_{III}$ for the warrant m_w^*. According to the Forking Lemma, by rewinding $\mathcal{A}_{III}$ and providing a new hash value of $h(h(m_w^*\|K_o^*)\|m^*\|K_p^*)$, $\mathcal{B}$ can obtain another valid signature $\hat{\sigma}^* = (K_o^*, K_p^*, \hat{\sigma}_p^*, \hat{s}^*))$ for (m_w^*, m^*). Then $\mathcal{B}$ can derive

$$S_p^* = (\sigma_p^* - \hat{\sigma}_p^*)/(h^* - \hat{h}^*) \bmod q$$

where h^* and $\hat{h}^*$ are the hash values for $h(m_w^*\|K_o^*)\|m^*\|K_p^*$ in the two executions. Finally, $\mathcal{B}$ can compute $x^* = S_p^* - \sigma_o^* \bmod q$ and solve the DLP.

6 Conclusion

In this paper, we presented a formal security model and an efficient construction of k-time proxy signature scheme. Our model has considered different types of potential adversaries against a k-time proxy signature scheme, and is to date the first complete formal security model for such schemes. We then presented a practical k-time proxy signature scheme based on the Schnorr signature and verifiable secret sharing. One interesting feature of our scheme is that the proxy signer's secret key can be discovered by the public if the proxy signer misbehaves. We also provided formal security proofs to demonstrate that the proposed scheme is provably secure in the proposed security model. We leave the problem of constructing a secure k-time proxy signature scheme without random oracles as our future work.

References

1. Awasthi, A.K., Lal, S.: ID-based ring signature and proxy ring signature schemes from bilinear pairings. arXiv preprint cs/0504097 (2005)
2. Awasthi, A.K., Lal, S.: Proxy blind signature scheme. Transaction on Cryptology 2(1), 5–11 (2005)
3. Bicakci, K., Tsudik, G., Tung, B.: How to construct optimal one-time signatures. Computer Networks 43(3), 339–349 (2003)
4. Chen, X., Zhang, F., Kim, K.: ID-based multi-proxy signature and blind multisignature from bilinear pairings. Proceedings of KIISC 3, 11–19 (2003)
5. Choi, C.-J., Kim, Z., Kim, K.: Schnorr signature scheme with restricted signing capability and its application. In: Proc. of CSS (2003)
6. Foster, I.T., Kesselman, C., Tsudik, G., Tuecke, S.: A security architecture for computational grids. In: ACM Conference on Computer and Communications Security, pp. 83–92 (1998)
7. Herranz, J., Sáez, G.: New identity-based ring signature schemes. In: López, J., Qing, S., Okamoto, E. (eds.) ICICS 2004. LNCS, vol. 3269, pp. 27–39. Springer, Heidelberg (2004)

8. Hong, X., Chen, K.: Secure multiple-times proxy signature scheme. Computer Standards and Interfaces 31(1), 19–23 (2009)
9. Hwang, J.Y., Kim, H.-J., Lee, D.H., Lim, J.I.: Digital signature schemes with restriction on signing capability. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, pp. 324–335. Springer, Heidelberg (2003)
10. Kim, S., Park, S., Won, D.: Proxy signatures, revisited. In: Han, Y., Quing, S. (eds.) ICICS 1997. LNCS, vol. 1334, pp. 223–232. Springer, Heidelberg (1997)
11. Kim, W.-H., Yoon, H., Cheon, J.H.: Metered signatures: How to restrict the signing capability. Journal of Communications and Networks 12(3), 201–208 (2010)
12. Lee, B., Kim, H., Kim, K.: Strong proxy signature and its applications. In: Proc. of SCIS, pp. 603–608 (2001)
13. Liu, J., Huang, S.: Identity-based threshold proxy signature from bilinear pairings. Informatica 21(1), 41–56 (2010)
14. Mambo, M., Usuda, K., Okamoto, E.: Proxy signatures for delegating signing operation. In: Proceedings of the 3rd ACM Conference on Computer and Communications Security, CCS 1996, pp. 48–57 (1996)
15. Mehta, M., Harn, L.: Efficient one-time proxy signatures. In: IEE Proceedings of the Communications, vol. 152, pp. 129–133. IET (2005)
16. Clifford Neuman, B.: Proxy-based authorization and accounting for distributed systems. In: ICDCS, pp. 283–291 (1993)
17. Pedersen, T.P.: Distributed provers with applications to undeniable signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 221–242. Springer, Heidelberg (1991)
18. Pieprzyk, J., Wang, H., Xing, C.: Multiple-time signature schemes against adaptive chosen message attacks. In: Matsui, M., Zuccherato, R.J. (eds.) SAC 2003. LNCS, vol. 3006, pp. 88–100. Springer, Heidelberg (2004)
19. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 387–398. Springer, Heidelberg (1996)
20. Wang, G.: Designated-verifier proxy signature schemes. In: Sasaki, R., Qing, S., Okamoto, E., Yoshiura, H. (eds.) SEC. IFIP AICT, vol. 181, pp. 409–423. Springer, Heidelberg (2005)
21. Wang, T., Wei, Z.: One-time proxy signature based on quantum cryptography. Quantum Information Processing 11(2), 455–463 (2012)
22. Wei, B., Zhang, F., Chen, X.: Ring proxy signatures. Journal of Electronics (China) 25(1), 108–114 (2008)
23. Wu, W., Mu, Y., Susilo, W., Seberry, J., Huang, X.: Identity-based proxy signature from pairings. In: Xiao, B., Yang, L.T., Ma, J., Muller-Schloer, C., Hua, Y. (eds.) ATC 2007. LNCS, vol. 4610, pp. 22–31. Springer, Heidelberg (2007)
24. Xu, J., Zhang, Z., Feng, D.: ID-based proxy signature using bilinear pairings. In: Chen, G., Pan, Y., Guo, M., Lu, J. (eds.) ISPA-WS 2005. LNCS, vol. 3759, pp. 359–367. Springer, Heidelberg (2005)
25. Yu, Y., Mu, Y., Susilo, W., Sun, Y., Ji, Y.: Provably secure proxy signature scheme from factorization. Mathematical and Computer Modelling 55, 1160–1168 (2012)
26. Zhang, F., Kim, K.: Efficient ID-based blind signature and proxy signature from bilinear pairings. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, pp. 312–323. Springer, Heidelberg (2003)
27. Zhang, F., Safavi-Naini, R., Lin, C.-Y.: New proxy signature, proxy blind signature and proxy ring signature schemes from bilinear pairing. IACR Cryptology ePrint Archive, 104 (2003)
28. Zhang, K.: Threshold proxy signature schemes. In: Okamoto, E. (ed.) ISW 1997. LNCS, vol. 1396, pp. 282–290. Springer, Heidelberg (1998)