Another Efficient Proxy Signature Scheme in the Standard Model* 

JIAN-HONG ZHANG AND JIAN MAO+  
Institute of Image Processing and Pattern Recognition 
North China University of Technology 
Beijing, 100144 P.R. China 
E-mail: jhzhangs@163.com
+School of Electronic and Information Engineering 
Beihang University 
Beijing, 10083 P.R. China 
E-mail: maojian@buaa.edu.cn 

As an important delegation technology, proxy signature allows an original signer to delegate her signing capability to a proxy signer and the proxy signer can produce a signature on behalf of the original signer. Presently, the length of proxy signature is a sum of lengths of two signatures in most of proxy signature schemes. It limits some applications for proxy signature. In this paper, we propose an efficient short proxy signature scheme without random oracle model based on Zhang et al.’s signature scheme. And the scheme is proven secure in the standard model and the security of the scheme is related to the $k + 1$-Square Roots Assumption. Compared with Huang et al.’s scheme, our scheme has several advantages over Huang et al.’s scheme in terms of the size of public key and computational costs of generation and verification of proxy signature. It is very suitable for mobile device.

Keywords: mobile agent, proxy signature, the $k + 1$-square roots problem, security proof, standard model

1. INTRODUCTION

In Mobile Ad hoc Networks, permanent connections between customers and servers are unnecessary and impracticable. To ensure service availability to the customers distributed in the whole networks, the server must delegate his rights to some other parties in the systems, such as mobile agents. A good way to realize this delegation is proxy signature technology.

The notion of proxy signature scheme is introduced by Mambo et al. in 1996 [1]. A proxy signature scheme allows an entity, called original signer, to delegate his signing capability to one or more entities, called proxy signer. Since it was proposed, the proxy signature schemes have been suggested for use in many applications [2, 6-8], particularly in distributed computing where delegation of rights is quite common. Examples discussed in the literature include distributed systems, Grid computing, mobile agent applications, distributed shared object systems, global distribution networks, and mobile communications. And to adapt different situations, many proxy signature variants have been produ-

Received March 31, 2010; revised July 26, 2010; accepted September 16, 2010. 
Communicated by Wen-Guey Tzeng.

* This paper was supported by Natural Science Foundation of China (No. 60703044), the New Star Plan Project of Beijing Science and Technology (No. 2007B001), and The Beijing Natural Science Foundation Program and Scientific Research Key Program of Beijing Municipal Commission of Education (No. KZ200810009005).
ced, such as one-time proxy signature, proxy blind signature, proxy verifiably encrypted signature, multi-proxy signature, proxy signcryption, and so on. Since the proxy signature appeared, it has attracted many researchers’ great attention. Based on the delegation type, proxy signature schemes are divided into full delegation, partial delegation and delegation by warrant. According to whether the original signer knows the secret key of proxy signer, proxy signatures can also be classified as proxy-unprotected and proxy-protected schemes. In a proxy-protected scheme the original signer cannot forge a proxy signer to produce a proxy signature. It means that proxy signature can be produced only by the proxy signer. Thus we can clearly distinguish the rights and responsibilities between the original signer and the proxy signer.

Provable security is a basic requirement for a signature scheme. Until now, almost all of proxy signature schemes are only proven secure in the random oracle model which was introduced by Bellare and Rogaway in [20]. In the model, hash function is regarded as a random generator. Thus, there exist constructions of various cryptographic schemes [3-5] provably secure in the random oracle model, but for which no instantiation of the random oracle yields a secure scheme in the standard model. As a consequence, a central line of research in modern cryptography is designing efficient schemes provably secure in the standard model. Recently, Huang et al. [10] proposed a proxy signature scheme without random oracle model based on Waters’ signature scheme. Because Waters’ signature scheme is not very efficient in terms of the size of public key and computational cost of producing signature, the drawbacks of Huang et al.’s scheme are the relatively large size of its public parameters and the complicated computational cost inheriting from Waters’ approach. It is an open problem to construct a short efficient proxy signature scheme without random oracle model.

Being inspired with the problems above, in this paper, based on Zhang et al.’s short signature scheme [24], we propose an efficient proxy signature scheme without random oracle model. At the same time, we also show that the security of the scheme is tightly related to the $k + 1$-Square Root Assumption in the standard model. Compared with Huang et al.’s proxy signature scheme, our scheme has several advantages over Huang et al.’s scheme in terms of the size of public key and computational costs of producing proxy signature; (1) the shorter size of proxy signature; (2) less computational cost in the signing phase and verification; (3) the shorter size of public key.

The rest of the paper is organized as follows. In section 2, we review some preliminary requirements and security assumptions throughout the paper. In section 3, we describe the formal models of our proxy signature scheme and our scheme is proposed in section 4. In section 5, security analysis and efficiency analysis of our scheme is given. Finally, we draw this paper.

2. PRELIMINARIES

In this section, we briefly review the basic definition and properties of the bilinear pairings. Bilinear pairings have been found to be very useful in various applications in recent years and have us to construct new cryptographic primitives. We recall some notations [6, 23] which are related to bilinear pairings.

Let $G_1$ be a cyclic multiplicative group generated by the generator $g$, whose order is
a prime $q$, and $G_2$ be a cyclic multiplicative group of the same prime order $q$. We assume that the discrete logarithm problem (DLP) in both $G_1$ and $G_2$ is hard. An admissible pairing $e: G_1 \times G_1 \rightarrow G_2$, which satisfies the following three properties:

1. Bilinearity: If $u, v \in G_1$ and $a, b \in \mathbb{Z}^*_q$, then $e(u^a, v^b) = e(u, v)^{ab}$;
2. Non-degenerate: There exists a $g \in G_1$ such that $e(g, g) \neq 1$;
3. Computable: If $u, v \in G_1$, any one can efficiently compute $e(u, v) \in G_2$ in polynomial time.

We note the modified Weil and Tate pairings associated with supersingular elliptic curves are examples of such admissible pairings. For security of a signature, the strongest security notion was defined by Goldwasser, Micali and Rivest [12]. The security of the scheme discussed in this paper is based on the following strongest security assumption.

**Definition 1** Existential unforgeability under adaptive chosen message attack of a signature scheme [12].

Let $DSC = (G, K, S, V)$ is a digital signature scheme. We consider a polynomial probabilistic algorithm $F$ that is given input a public key $pk$ and able to access to a hash oracle $OH(\cdot)$ and a signing oracle $Os(sk, \cdot)$. Let define $\mathsf{CMA}_F$ for $(t, q_H, q_{sig}, \varepsilon)$-break a digital signature scheme if after running in at most $t$ steps, making at most $q_H$ adaptive queries to $OH(\cdot)$ and at most $q_{sig}$ adaptively chosen message queries to $Os(sk, \cdot)$, $F$ outputs a valid forgery $(M, \sigma)$ on some new message $M$ (i.e. this message was not queried to the signing oracle.) with $\mathsf{CMA}_F \geq \varepsilon$. DS is said to be $(t, q_H, q_{sig}, \varepsilon)$-secure if only if no forger can $(t, q_H, q_{sig}, \varepsilon)$ break it. DS is said to be secure against existential forgery under adaptive chosen message attack if $\varepsilon$ is negligible in the security parameter $k$ and DS is said to be $(t, q_H, q_{sig}, \varepsilon)$-secure.

**Definition 2** $(k + 1$-SRP) The $(k + 1$-Square Roots Problem in $(G_1, G_T)$ is as follows: for an integer $k$, and $x \in \mathbb{Z}_q, g \in \mathbb{Z}_q, g \in G_1$, given

$$\{g, \alpha = g^\varepsilon, h_1, h_2, \ldots, h_k \in \mathbb{Z}_q, g^{(x+h)^{1/2}}, \ldots, g^{(x+h)^{1/2}}, \}$$

compute $g^{(x+h)^{1/2}}$ for some $h \notin \{h_1, \ldots, h_k\}$. We say the $(k + 1$-SRP is $(t, \varepsilon)$-hard if for any $t$-time adversary $A$, we have

$$\Pr \left[ A(g, \alpha = g^\varepsilon, g^{(x+h)^{1/2}}, \ldots, g^{(x+h)^{1/2}} | x \in \mathbb{Z}_q, g \in G_1, h_1, h_2, \ldots, h_k \in \mathbb{Z}_q \right] \leq \varepsilon.$$
Definition 3 (\(k + 1\)-SR assumption) We say that \((k + 1, t, \varepsilon)\)-SR assumption holds in groups \((G_1, G_2)\) if no \(t\)-time algorithm has advantage at least \(\varepsilon\) in solving the \(k + 1\)-SR problem in \((G_1, G_2)\), i.e., \(k + 1\)-SRP is \((t, \varepsilon)\)-hard in \((G_1, G_2)\).

Remarks: \(k + 1\)-Square Roots Problem is not a well studied problem and we are uncertain of its difficulty. The security of our proposed scheme relies on \(k + 1\)-SR security assumptions.

2.1 Proxy Signature Scheme

A proxy signature scheme consists of three entities: an original signer, a proxy signer and a verifier.

- Setup: The probabilistic generation algorithm that takes as input a security parameter \(l\), and outputs system parameters: \(\text{param}\). The original signer and proxy signer produce their secret-public key pair \((sk_o, pk_o)\) and \((sk_p, pk_p)\), respectively. Note that, in fact, Setup phase consists of parameters generation algorithm (ParamGen) and key generation algorithm (KeyGen).

- Delegation algorithm DL: The algorithm takes as the input the secret key \(sk_o\) of an original signer and a warrant \(W\), where the warrant \(W\) contains the identity (ID) of proxy signer and, possibly, restrictions on the message the proxy signer is allowed to sign. Finally, output the proxy signing key \(sp\).

- Proxy Signing Algorithm PS: The algorithm takes input the proxy signer’s proxy signing key \(sp\), the proxy signer’s public key \(pk_p\) and the message \(M\), and outputs the proxy signature \(\delta_p\) of the message \(M\).

- Proxy signature Verification PV: A deterministic algorithm PV takes input \((pk_o, pk_p, M, W, \delta_p)\), and outputs a bit, where \(W\) is a warrant of warrant. We say that \(\delta_p\) is a valid proxy signature for \(M\) if

\[
PV(pk_o, pk_p, M, \delta_p, W) = 1,
\]

otherwise outputs false \(\bot\).

2.2 Security Requirements of Proxy Signature

Since Mambo et al. introduced the conception of proxy signature, many security requirements of proxy signature are added continually to satisfy the requirement in different situations. In order to make a proxy signature scheme fairer to the responsibility of the original signer and the proxy signer in some cases. The security requirements of a secure proxy signature scheme are described as follows,

- Verifiability: For the proxy signature, a verifier can be convinced of an original signer’s agreement on the signed message.

- Strong unforgeability: A proxy signer can create a valid proxy signature on behalf the original signer. However, the original signer and any third party can not generate a valid proxy signature in the name of proxy signer.
- Strong identifiability: For a proxy signature, anyone can determine the identity of the corresponding proxy signer.
- Strong undeniability: Once a proxy signer generates a valid proxy signature on behalf of the proxy signer, the proxy signer can’t deny his signature generation against anyone.
- Prevention of misuse: It should be confident that proxy key pair can’t be used for other purposes. In the case of misuse, the responsibility of proxy signature should be determined explicitly.

Where Unforgeability is the most important property in a proxy signature. It denotes that only delegated proxy signer can generate a valid proxy signature and original signer cannot produce a valid proxy signature on behalf of proxy signer. In fact, unforgeability includes the undeniability and prevention of misuse.

According to the model defined in [10, 11], we divide the potential adversary into three attack types:

1. Type I: In this attack type, an adversary $A_I$ only has the public keys of original signer and proxy signer.
2. Type II: In this attack type, an adversary $A_{II}$ has the public keys of original signer and proxy signer, and it has also the secret key of the proxy signer.
3. Type III: In this attack type, an adversary $A_{III}$ has the public keys of original signer and proxy signer, it has also the secret key of original signer.

Obviously, we know that if a proxy signature scheme is secure against Type II (or Type III) adversary, the scheme is also secure against Type I adversary. In the following security model, we only consider Type II adversary and Type III adversary.

**Existential unforgeability against adaptive $A_{II}$ adversary:** Roughly speaking, the existential unforgeability of a short proxy signature scheme under adaptive $A_{II}$ attacker requires that it is difficult for a user to forge a valid proxy signature under a warrant $W$ by the following game between a challenger $C$ and the adversary $A_{II}$.

1. $C$ runs Setup algorithms, and produces proxy signer’s secret-public ($sk_p, pk_p$) and original signer’s public key $pk_o$. Then its resulting system parameters and the secret key $sk_p$ of proxy signer are given to $A_{II}$.
2. $A_{II}$ can issue the following queries:
   (a) Delegation queries: Proceeding adaptively, when $A_{II}$ requests the delegation with a warrant $W$, $C$ runs the Delegation algorithm $DG$ to obtain proxy signing key $sp$ and sends it to the $A_{II}$.
   (b) ProxySign queries: Proceeding adaptively, $A_{II}$ can request the proxy signature on any message $M$ of his choice. In response, $C$ runs Delegation algorithm $DG$ to generate the delegation on the warrant $W$, where $M$ must belong to the admission range of the warrant $W$. Then $C$ runs the ProxySign algorithm to obtain the proxy signature $\delta$ on message $M$ and returns $(W, \delta)$ to the adversary $A_{II}$.
3. **Outputs:** Finally, $A_{II}$ outputs a forgery proxy signature $\delta^*$ with a warrant $W'$ and the message $M'$ such that
JIAN-HONG ZHANG AND JIAN MAO

(a) the adversary cannot obtain the delegation of $W'$ through delegation queries.
(b) $(W', M')$ has never been queried on the Proxysign queries.
(c) $\delta'$ is a valid signature on message $M'$ and $M'$ belongs to the admission of $W'$.

Compared with the model defined in [2, 9], an important refinement is $A_{II}$ can adaptively request the ProxySign queries with message $M'$ under the warrant $W$. The success probability of an algorithm $A_{II}$ wins the above game is defined as $\text{Succ}_{A_{II}}$.

**Definition 5** We say a Type II adversary $A_{II}$ can break a proxy signature scheme if $A_{II}$ runs in time at most $t$, $A_{II}$ makes at most $q_d$ delegation queries and at most $q_s$ ProxySign queries and $\text{Succ}_{A_{II}}$ is at least $\epsilon$.

**Existential unforgeability against adaptive $A_{III}$ adversary:** Roughly speaking, the attack shows that a proxy signature is only produced by proxy signer, even if the original signer can not also produce a proxy signature. The existential unforgeability of a proxy signature scheme with warrant under a Type III attacker requires that it is difficult for the original signer to output a valid proxy signature by the following game between the challenger $C$ and the adversary $A_{III}$.

1. $C$ runs Setup algorithms, and produces original signer’s secret-public $(sk_0, pk_0)$ and proxy signer’s public key $pk_p$. Then its resulting system parameters and the secret key $sk_0$ of original signer are given to $A_{III}$.

2. $A_{III}$ can issue the following queries: ProxySign queries: Proceeding adaptively, $A_{III}$ can request the proxy signature on any message $M$. In response, $C$ runs Delegation algorithm $DG$ to generate the delegation on the warrant $W$, where $M$ must belong to the admission range of the warrant $W$. Then $C$ runs the ProxySign algorithm to obtain the proxy signature $\delta$ on message $M$ and returns $(W, \delta)$ to the adversary $A_{III}$. Note that the adversary $A_{III}$ doesn’t need to request delegation queries, since it has the secret key of original signer.

3. **Outputs:** Finally, $A_{III}$ outputs a forgery proxy signature $\delta'$ with a warrant $W'$ and the message $M'$ such that
   (a) $M'$ has never been requested as one of the Proxysign queries.
   (c) $\delta'$ is a valid signature on message $M'$ and $M'$ belongs to the admission of $W'$.

The success probability of an algorithm $A_{III}$ wins the above game is defined as $\text{Succ}_{A_{III}}$.

**Definition 6** We say a Type II adversary $A_{II}$ can break a proxy signature scheme if $A_{II}$ runs in time at most $t$, $A_{II}$ makes at most $q_d$ delegation queries and at most $q_s$ ProxySign queries and $\text{Succ}_{A_{II}}$ is at least $\epsilon$.

**3. OUR PROXY SIGNATURE SCHEME**

In this section, we will propose an efficient proxy signature scheme in the standard model. Our scheme is based on Zhang et al.’s short signature scheme [24]. Our scheme consists of the following steps:
In the setup phase, **ParamGen** algorithm and **KeyGen** algorithm is computed as follows,

**ParamGen**: Let $G_1, G_T$ be two cyclic groups of order $q$ which is a prime number and $g$ is the generator of $G_1$. $e$ denotes the bilinear pairing map $G_1 \times G_1 \to G_T$. The master public parameters are $(g, G_1, G_T, e, q)$.

**KeyGen**: The original signer Alice randomly chooses $x_a, y_a \in \mathbb{Z}_q$ to compute the corresponding public key $u_a = g^{x_a}$ and $v_a = g^{y_a}$. Similarly, for the proxy signer Bob, he also randomly selects $x_b, y_b \in \mathbb{Z}_q$ to produce the corresponding public key $u_b = g^{x_b}$ and $v_b = g^{y_b}$.

**Delegation**: Let $W$ denote a delegated warrant which includes proxy signer’s identity and deadline, and so on. To produce a delegation of warrant $W$, the original signer Alice computes as follows,

- Randomly choose $r \in \mathbb{R} \mathbb{Z}_q$ to compute $x_a + Wy_a + r$. If $x_a + Wy_a + r$ is not a quadratic residue modulo $q$, and then try again with a different value $r$.
- Then, compute $\delta = g^{(x_a + Wy_a + r)^2}$ and send $(\delta, r)$ to proxy signer Bob.
- Upon receiving $(\delta, r)$, proxy signer verifies whether the following equation holds. If it holds, then $(\delta, r)$ is acted as the signing key of proxy signer.

**ProxySign**: Let $M$ be a 160-bit message in the admission range of warrant $W$. Otherwise, we can adopt a suitable collision resistant hash function to hash the message to 160bits. To generate a signature $\operatorname{Sig}$ on the message $M$ with $(\delta, r)$ and secret key $(x_b, y_b)$, the proxy signer computes as follows,

1. randomly choose $r_b \in \mathbb{Z}_q$ to compute $x_b + My_b + r_b$. If $x_b + My_b + r_b$ is not a quadratic residue modulo $q$, then we try again with another $r_b \in \mathbb{R} \mathbb{Z}_q$.
2. then compute $\beta = g^{(x_b + My_b + r_b)^2}$.
3. the resultant proxy signature on message $M$ is $(\beta, W, r_b, r)$.

**Verify**: Given a proxy signature $(\beta, W, r_b, r)$ on message $M$, a verifier first checks whether $M$ belongs to the admission ranger of $W$. If it is valid, then it verifies as follows,

$$e(\beta, \beta) = e(u_a, v_a^M g^r, u_b, v_b^N g^h). \tag{1}$$

If the above Eq. (1) holds, then the result returns **True**; otherwise, the result returns **False**.

### 4. Security Analysis

#### 4.1 Correctness

Clearly, the correctness can be easily verified by the following equations.
4.2 Analysis

In the following, we will provide security analysis of the proposed proxy signature scheme and show that the scheme is secure in the standard model.

**Theorem 1**  If there exists an adversary $A_{II}$ can $(t, q_d, q_s, \tau, \varepsilon)$ break the proposed proxy signature scheme, then there exists another algorithm $B$ who can make use of the adversary $A_{II}$ to solve the $k + 1$-SR problem in group $(G_1, G_T)$ with the probability

$$\varepsilon' > \frac{2}{2} q_d + q_s.$$  

Where $q_d$ denotes at most times of asking proxy signing queries, $q_s$ be at most times of asking delegation queries.

**Proof:** Assume there is a $(t, q_d, q_s, \varepsilon)$-adversary $A_{II}$ exists. We are going to construct another PPT $B$ who makes use of $A_{II}$ to solve the $k + 1$-SR problem with probability at least $\varepsilon'$ and in time at most $t'$.  

Let us recall the $k + 1$-SR problem, given a $k + 1$-SR problem instance $\{g, x = g^y, h_1, h_2, \ldots, h_{k+1} \in Z_q, g^{(x+h_1)^{k+2}}, \ldots, g^{(x+h_{k+1})^{k+2}}\}$. Its goal is to compute $g^{(x+h)^{k+2}}$ for some a value $h \not\in \{h_1, \ldots, h_{k+1}\}$, where $k + 1 > q_d + q_s$. In order to use $A_{II}$ to solve this problem, $B$ needs to simulate a challenger and the oracles (Delegation oracle and proxy signing oracle) for the adversary $A$. To efficiently simulate their interactive steps, we distinguish two types of forgers. Let $\{G_1, G_T, g, u, v, u_0, v_0\}$ be public parameters which are given to the adversary $A_{II}$. When this adversary $A_{II}$ asks for delegation oracle on warrants $(W_1, \ldots, W_{q_d})$, $(r, \delta)$ is responded on these warrants $W_i$ for $i = 1, 2, \ldots, q_d$. Let $h_i = W_i y_a + r_i$ and denote two types of forger $A_{II}$ as follows,

- **Type 1:** $A_{II}$ which makes query for some warrant satisfying $W_i = -x_a$ or outputs a forgery where $W \cdot y_a + r \not\in \{h_1, \ldots, h_{q_d}\}$.
- **Type 2:** $A_{II}$ which never makes any query for a warrant which satisfies $W_i = -x_a$, and outputs a forgery where $W \cdot y_a + r \in \{h_1, \ldots, h_{q_d}\}$.

In the following, we describe their interactive steps.

**Setup:** To simulate the game, $B$ chooses a random element $y_a \in Z_q$ to compute $v_a = g^{y_a}$. And let $(u_0 = (\alpha)^k(g^{x_a})^{k+2}, v_0 = \alpha^{1-k}(g^{x_a})^{k+2})$ be original signer’s public key, where $\alpha$ is the instance of the above $k + 1$-SR problem and $k' \in \{0, 1\}$. Then $B$ selects two integers $x_b$, $y_b \in Z_q$ to compute proxy signer’s public key $u_b = g^{x_b}$ and $v_b = g^{y_b}$. If $k = 1$, then $B$ sends original signer’s public key $(u_0, v_0 = g^{y_a})$, proxy signer’s public key $(u_b, v_b)$ and proxy signer’s secret key $(x_b, y_b)$ to the adversary $A_{II}$; if $k = 0$, then $B$ sends original signer’s
ANOTHER EFFICIENT PROXY SIGNATURE SCHEME IN THE STANDARD MODEL

public key \((u_a = g^{v_a}, v_a = \alpha)\), proxy signer’s public key \((u_b, v_b)\) and proxy signer’s secret key \((x_b, y_b)\) to the adversary \(A_{II}\).

**Delegation Oracle:** When \(A_{II}\) issues a delegation query with warrant \(W\), to respond these delegation queries, \(B\) maintains a list \(h\)-list which is initially empty and a counter \(l\) which is initially set to be 0.

1. Upon receiving a delegation query for warrant \(W_i\), \(B\) increments \(l\) by one, and checks whether the equation \(g^{-W_i} = u_a\) holds. If so, then it means that the relation \(\alpha = g^{W_i}\) holds, thus \(B\) can compute \((h, g^{(h + x_i)\frac{1}{2}})\) where \(h \in \{h_1, \ldots, h_q\}\).
2. Otherwise, if \(k = 1\), \(B\) sets \(r_i = h_i - W_i v_a\). When \(r_i = 0\), \(B\) reports failure and aborts it.
   Otherwise, \(B\) returns \((\delta_i, r_i, \beta_i)\) to the adversary \(A_{II}\). If \(k = 0\), \(B\) sets \(r_i = W_i h_i - y_a \in Z_q\) and checks whether \(r_i = 0\) holds, if holds, then \(B\) reports failure and aborts it.
3. Finally, add \((W_i, r_i, H_i = v_a W_i r_i, \delta_i)\) in the \(h\)-list.

**ProxySign Oracle:** Suppose \(A_{II}\) issues a ProxySign query on message \(m\) under the warrant \(W\), \(B\) responses as follows,

1. Firstly, \(B\) checks whether \((W, *, *, *)\) exists in the \(h\)-list. If so, then \(B\) returns this tuple \((W, r_i, *, \delta_i)\), and randomly chooses \(r_{bi} \in Z_q\) to compute \(\beta_i = \delta_i^{(x_b + m_i y_b + r_{bi})\frac{1}{2}}\). Note that the key pair \((x_b, y_b)\) of proxy signer is known to \(B\).
2. Otherwise, \(B\) makes a delegation oracle on warrant \(W_i\) to obtain \((W_i, r_i, *, \delta_i)\). Then produce a proxy signature by the foregoing step.
3. Finally, return proxy signature \((\beta_i, r_{bi}, r_i)\) on message \(m\) to the adversary \(A_{II}\).

**Output:** Finally, the adversary \(A_{II}\) outputs a valid proxy signature \((\beta, r_{*}, r_{*})\) on message \(m\) under the warrant \(W\) such that

1. \(m\) belongs to the admission range of \(W\);
2. \(W\) must never been queried for delegation oracle;
3. \(\beta\) is a valid proxy signature which satisfies the verifying equation.

Since \(B\) know secret key \((x_b, y_b)\) of proxy signer, it can compute as follows,

\[\delta = (\beta^{x_b + m_i y_b + r_{bi}})^{\frac{1}{2}}.\]

Thus, we have the following relation

\[e(\delta, \delta') = e(u_a v_a^{\delta'}, g^{\delta'}, u_b^{(x_b + m_i y_b + r_{bi})\frac{1}{2}})^{\frac{1}{2}} = e(u_a v_a^{\delta'} g^{\delta'}, g).\]  

Let \(H' = v_a^{\delta'} g^{\delta'}\), according to the above two types of the adversary \(A_{II}\), we want to define the following events:

**F1:** (Type 1 of adversary \(A_{II}\)) No tuple of the form \((*, *, H', *)\) appears in the \(h\)-list.

**F2:** (Type 2 of adversary \(A_{II}\)) At least one tuple \((W_i, r_i, H_i, \delta_i)\) which satisfies \(H_i = H'\) appears in the \(h\)-list.
Denote $E_1$ be the event $k = 1$ and $E_2$ be the event $k = 0$. We know that $A_{II}$ success in the above game if and only if $(E_1 \land F_1) \lor (E_2 \land F_2)$ happens.

**Case 1:** If $u_a = g^{-\omega}$, then it means $A_{II}$ can recover the secret key of its challenger. Thus SR-problem can be solved. When $W' y_a + r' \in \{h_1, h_2, \ldots, h_q\}$ holds, the forged signature $(\beta', r_a', r_j')$ is valid, it should satisfy the above Eq. (2). Let $h' = W' y_a + r'$. Then $(h', \delta')$ is a new solution of SR-problem.

**Case 2:** In the case, the relation $v_a = \alpha = g^\omega$ holds and there exists a pair $v_a' g^{r} = v_a' g^{r'}$. Since $(W', r') \neq (W, r)$, otherwise it does not satisfy the condition in the **Output** phase. It means that $W' \neq W$ and $r' \neq r$. Therefore, $B$ can compute

$$x = \frac{r_j - r^*}{W_j - W^*},$$

which is the secret key of its challenger. It also means that SR-problem is solved.

Now, we have to access $B$’s probability of success. Since $E_1$ and $F_1$ are independent with uniform distribution, $Pr[E_1 \lor E_2] = 1$ and $Pr[F_1 \lor F_2] = 1$, the probability that $A_{II}$ succeeds is $Pr[(E_1 \land F_1) \lor (E_2 \land F_2)] = 1/2$.

Next we bound the probability that $B$ does not abort. From the above game of $B$, we know that $A_{II}$ aborts if

- In $E_1 \land F_1$, if and only if $r_j = 0$ in the delegation phase and proxy signing phase. For a given $y_a$, it appears with probability at most $(q_d + q_s)/q$.
- In $E_2 \land F_2$, if and only if $r_j = 0$ in the delegation phase and proxy signing phase. For a given $y_a$, it appears with probability at most $(q_d + q_s)/q$.

Thus, $B$ succeeds with probability at least

$$\frac{\varepsilon}{2} - \frac{q_d + q_s}{q},$$

where $\varepsilon$ is a probability to produce a valid forgeable proxy signature. 

**Theorem 2** If there exists an adversary $A_{III}$ can $(t, q_d, q_s, \tau, \varepsilon)$ break the proposed proxy signature scheme, then there exists another algorithm $B$ who is able to use the adversary $A_{III}$ to solve the $k + 1$-SR problem in group $(G_1, G_2)$ with the probability

$$\varepsilon' > \frac{\varepsilon}{2} - 2q_d + q_s.$$ 

Where $q_s$ denotes at most times of asking proxy signing queries, $q_d$ be at most times of asking delegation queries.

**Proof:** It is similar to the proof of Theorem 1.

Let us review the $k + 1$-SR problem, given a $k + 1$-SR problem instance
\[ \{ g, \alpha = g^r, h_1, h_2, \ldots, h_{k+1} \in \mathbb{Z}_q, g^{(x+y)^i} \} \]

Its goal is to compute \( g^{(x+y)^i} \) for some a value \( h \in \{ h_1, \ldots, h_{k+1} \} \), where \( k + 1 > q_d + q_c \). In order to use \( A_{\text{III}} \) to solve the \( k + 1 \)-SR problem, \( B \) needs to simulate a challenger and the oracles (Delegation oracle and proxy signing oracle) for the adversary \( A_{\text{III}} \). To efficiently simulate their interactive steps, we distinguish two types of forgers. Let

\[ \{ G_1, G_2, q, u_a, v_a, u_b, v_b \} \]

be public parameters which are given to the adversary \( A_{\text{III}} \). When this adversary \( A_{\text{III}} \) asks for Proxy Signing oracle on message \( m \), \((r_i, r_b, \delta)\) is responded for \( i = 1, 2, \ldots, q \). Let \( h_i = m v_b + r_b \) and denote two types of forger \( A_{\text{III}} \) as follows:

- **Type 1:** \( A_{\text{III}} \) which makes query for some message satisfying \( m_1 = -x_0 \) or outputs a forgery where \( m \cdot y_b + r_b \notin \{ h_1, \ldots, h_{q_b} \} \).
- **Type 2:** \( A_{\text{III}} \) which never makes any query for a message \( m_i = -x_0 \), and outputs a forgery where \( m \cdot y_b + r_b \notin \{ h_1, \ldots, h_{q_b} \} \).

In the following, we describe their interactive steps.

**Setup:** To simulate the game, \( B \) chooses a random element \( y_b \in \mathbb{Z}_q \) to compute \( v_b = g^{y_b} \). And let \((u_b = (\alpha)^i (g^{y_b})^{i-1}, v_b = \alpha^{i-1} (g^{y_b})^{i-1})\) be proxy signer’s public key, where \( \alpha \) is the instance of the above \( k + 1 \)-SR problem and \( k' \in \{0, 1\} \). Then \( B \) selects two integers \( x_a, y_a \in \mathbb{Z}_q \) to compute original signer’s public key \( u_a = g^{x_a} \) and \( v_a = g^{y_a} \). If \( k = 1 \), then \( B \) sends proxy signer’s public key \((u_b = \alpha, v_b = g^{y_b})\), original signer’s public key \((u_a, v_a)\) and original signer’s secret key \((x_a, y_a)\) to the adversary \( A_{\text{III}} \); if \( k = 0 \), then \( B \) sends proxy signer’s public key \((u_b = g^{y_b}, v_b = \alpha)\), original signer’s public key \((u_a, v_a)\) and original signer’s secret key \((x_a, y_a)\) to the adversary \( A_{\text{III}} \).

**Delegation Oracle:** Because the adversary \( A_{\text{III}} \) possesses original signer’s secret key, Delegation Oracle is not needed to be queried.

**ProxySign Oracle:** Suppose \( A_{\text{III}} \) issues a ProxySign query on message \( m \) under the warrant \( W_i, B \) responses as follows,

1. Upon receiving a proxy signing query for message \( m \), \( B \) increments \( l \) by one and checks whether the equation \( g^{m_i} = u_b \). If so, then it means \( \alpha = g^{m_i} \), thus \( B \) can produce a pair \( (h_i, g^{(x+y)^i}) \) where \( h \in \{ h_1, \ldots, h_{q_b} \} \). Thus, the \( k + 1 \)-SR problem is solved.
2. Otherwise, when \( k = 1 \), \( B \) sets \( r_b = h_i - m v_b \). In the very unlikely event that \( r_b = 0 \), \( B \) selects another value from the remainder set \( \{ h_1, \ldots, h_{q_b} \} \) to obtain \( r_b \neq 0 \). Then randomly choose a \( r_i \in \mathbb{Z}_q \) to compute \( \beta_i = (g^{(k+1)x_i})^{(m_i + W_i y_i + r_i)/2} \). Finally, \( B \) returns

\[(r_i, r_b, W_i, \beta_i, H_i = v_b = g^{y_b})\]

in the \( h \)-list. When \( k = 0 \), \( B \) sets \( r_b = m v_b - y_b \in \mathbb{Z}_q \), if the computed \( r_b = 0 \), another value is selected from \( \{ h_1, \ldots, h_{q_b} \} \). Then randomly choose a \( r_i \in \mathbb{Z}_q \) to compute
\[
\beta_i = \left( g^{(h_{k+1})^{2j}} \right)^{\sqrt{m_i}} \left( x_{i} + W\beta_i + \eta_i \right)^{\frac{1}{2}}.
\]

Then, \( B \) returns \((r_i, r_\beta, W_i, \beta_i, H_i = v_i^{m_i} g^{b_i})\) in the \( H \)-list.

3. Finally, return proxy signature \((\beta, r_\beta, r_i)\) on message \( m_i \) to the adversary \( A_{III} \).

**Output:** Finally, the adversary \( A_{III} \) outputs a valid proxy signature \((\beta^*, r_\beta^*, r^*)\) on message \( m^* \) under the warrant \( W^* \) such that

1. \( m^* \) belongs to the admission range of \( W^* \);
2. \( m^* \) must never been queried for proxy signing oracle;
3. \( \beta^* \) is a valid proxy signature which satisfies the verifying equation.

Since \( B \) know secret key \((x, y)\) of original signer, it can compute as follows,

\[
\tau^* = (\beta^*)^{x_i + W_{x_i} + \eta_i}^{\frac{1}{2}}.
\]

Thus, we have the following relation

\[
e(\tau^*, \tau^*) = e(u_b v_{m}^{m_i} g^{b_i}, u_b v_{b}^{m_i} g^{b_i})^{x_i + W_{x_i} + \eta_i} = e(u_b v_{b}^{m_i} g^{b_i}, g).
\]

Let \( H^* = v_i^{m_i} g^{\alpha} \), according to the above two types of the adversary \( A_{III} \), we want to define the following events:

**F1:** (Type 1 of adversary \( A_{III} \)) No tuple of the form \((*, *, *, *, H^*)\) appears in the \( H \)-list.

**F2:** (Type 2 of adversary \( A_{III} \)) At least one tuple \((r_i, r_\beta, W_i, \beta_i, H_i)\) which satisfies \( H_i = H^* \) appears in the \( H \)-list.

Denote \( E_1 \) be the event \( k = 1 \) and \( E_2 \) be the event \( k = 0 \). We know that \( A_{III} \) success in the above game if and only if \((E_1 \land F1) \lor (E_2 \land F2) \) happens.

**Case 1:** If \( u_b = g^{m_i} \), then it means \( A_{III} \) can recover the secret key of its challenger. Thus SR-problem can be solved. When \( m y_b + r^* \notin \{h_1, h_2, \ldots, h_{qs}\} \) holds, the forged signature \((\beta^*, r_\beta^*, r^*)\) is valid, thus \((\tau^*, m^*, r^*)\) should satisfy the above Eq. (3). Let \( h^* = m y_b + r^* \). Then \((h^*, \tau^*)\) is a new solution of SR-problem.

**Case 2:** In the case, the relation \( v_b = \alpha = g^{\alpha} \) holds and there exists a pair \( v_i^{w_i} g^{\beta_i} = v_i^{w_i} g^{b_i} \). Since \((m^*, r^*) \neq (m_j, r_b)\), otherwise it does not satisfy the condition in the **Output** phase. It means that \( m \neq m_j \) and \( r^* \neq r_b \). Therefore, \( B \) can compute

\[
x = \frac{r_j - r^*}{m_j - m}.
\]

which is the secret key of its challenger. It also means that SR-problem is solved.

The analysis of \( B \)'s probability of success is similar to that of Theorem 1, for the limited space, we omit it here.
4.3 Efficiency Analysis

Here, we compare our scheme with Huang et al.’s scheme [10] in terms of signature size, public key size and computational costs of verifying and signing. Because Huang et al.’s scheme is also a proxy signature without random oracle. To fairly comparing, we include the following presentation, the notion $|G_1|$ denotes the bit length of an element in $G_1$, $P_m$ be scalar multiplication on the curve, $P_a$ be multiplication among elements in group $G_1$ and $e$ be pairings computation which is very expensive compared to summation and exponentiation and it determines efficiency of a scheme. Let $|q|$ be binary length of $q$ and $|G_1|$ be length of element in group $G_1$.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Size</th>
<th>Verification</th>
<th>Proxy signing</th>
<th>Size of PK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huang et al.’s scheme</td>
<td>$3</td>
<td>G_1</td>
<td>$</td>
<td>$5e + 2nP_a$</td>
</tr>
<tr>
<td>Our scheme</td>
<td>$1</td>
<td>G_1</td>
<td>+ 2</td>
<td>q</td>
</tr>
</tbody>
</table>

Let size be the length of signature, verification be verification computation, proxy signing be signing cost, size of PK be size of public key of proxy signer or original signer.

According to the following table, proxy signature in our scheme has the advantages over that of Huang et al.’s scheme in terms of the size of system parameters, computational cost of verification and proxy signing and the size of proxy signature.

**Strong undeniability:** According to the scheme above, we know that only the proxy signer can generate such a proxy signature. Therefore, he cannot deny his responsibility.

**Prevention of misuse:** In our scheme, the delegation signature and proxy signing key cannot be used to learn any of the private keys. Thereby, they cannot be used for other purposes than proxy signing and delegating signing capability. The compliance of these actions with their respective warrants is to be enforced by any entity verifying a proxy signature or being delegated a signing capability on behalf of some other entities.

**Signature Length:** A signature size in our proposed scheme only consists of one element $\beta$ in $G_1$ and two elements $(r_b, r)$ in $\mathbb{Z}_q$. When using a supersingular elliptic curve over finite field $\mathbb{F}_{p^n}$ with embedding degree $k = 6$ and the modified Weil pairing or Tate pairing [19, 20], the length of an element in $G_1$ and in $\mathbb{Z}_q$ can be approximately $\log_2 q$ bits, thus the total signature length is approximately $3\log_2 q$ bits.

According to the above proxy signing process, we know that $r_b$ in the proxy signature $(\beta, r, r_b)$ is a random number. Thus, we can make $r_b = r$ to reduce the size of proxy signature. But $x_b + My_b + r_b$ may not be a quadratic residue modulo $q$. To ensure $x_b + My_b + r_b$ be a quadratic residue modulo $q$, we can include a hash function $h(\cdot)$ to set $r_b = h(r)$, where $h(r) = h^{-1}(h(r))$ and $I$ is the first number which satisfies $x_b + My_b + h(r)$ to be a quadratic residue modulo $q$. Then the resultant proxy signature is $(\beta, r, i)$, where $i$ is a $|q|/2$ bits number. Since there are $(q - 1)/2$ quadratic residue numbers in $\mathbb{Z}_q$. Thus, the size of proxy signature is reduced to $1|G_1| + 3|q|/2$. 

Table 1. Comparison of our proposed scheme with Huang et al.’s scheme.
5. CONCLUSION

As a special signature type, proxy signature plays an important role in right of delegation. In this paper, we proposed an efficient proxy signature scheme without random oracle model based on Zhang et al.’s signature scheme. Then we show that the scheme is proven secure in the standard model and the security of the scheme is related to the $k + 1$ Square Roots Assumption. Compared with Huang et al.’s scheme, our scheme has several advantages over Huang et al.’s scheme in terms of the size of public key and computational costs of generation and verification of proxy signature. A prominent merit in our scheme is less computational costs in the signing phase. And our scheme is not malleable, while Huang et al.’s scheme is malleable.

REFERENCES

11. W. Wu, Y. Mu, W. Susilo, J. Seberry, and X. Huang, “Identity-based proxy signa-
Jian-Hong Zhang (张键红) received his Ph.D. degrees in Cryptography from Xidian University, Xi’an, Shanxi, in 2004 and his M.S. degree in Computer Software from Guizhou University, Guiyang, Guizhou, in 2001. He was engaging in postdoctoral research at Peking University from October 2005 to December 2007. He has been an Assistant Processor of College of Sciences, North China University of Technology, Beijing China, since 2001. His research interests include computer networks, cryptography, electronic commerce security, computer software.

Jian Mao (毛剑) received the B.S. degree in Electrical Engineering from Xidian University, Xi’an, Shanxi, in 1997, the M.S. degree in Computer Science from Xidian University Xi’an Shanxi, in 2001, and the Ph.D. degree in Cryptography from Xidian University, in 2004. She is a Lecture of the Department of Computer Science and Information Engineering of Beihang University, Beijing. Her areas of research include information security, cryptography, broadcast encryption, security in the wireless network, software security analysis.