User Efficient Randomized Chaum's Blind Signatures

Abstract

This manuscript presents a new randomized Chaum's blind signature scheme to reduce users' computation loads for the situations where their computation capabilities are limited such as smart-card customers and mobile clients. Comparing with the original scheme, the computations required for users are reduced by about 40% in general and more than 95% if we take a short key \( e = 3 \). In addition, the randomization and unlikeliness of the proposed scheme are examined.

Keywords: Blind signatures, Electronic cash, Electronic voting, Cryptography

1 Introduction

The concept of blind signatures was first introduced by Chaum [2] to prevent digital signatures from being forged and to protect the privacy of users. Based on the RSA cryptosystem, Chaum proposed the first blind signature scheme to achieve the unlikeliness property [2]. By means of the techniques of blind signatures, many anonymous electronic voting protocols [1, 5, 11] and untraceable electronic cash systems [3, 10, 14, 15] have been proposed.

In general, two kinds of roles, a signer and a group of users, participate in a blind signature protocol. A user blinds a message by performing an encryption-like process (or a blinding process) on the message, and then submits the blinded message to the signer to request the signer's signature on the blinded message. The signer signs on the blinded message by using its signing
function, and then sends the signing result back to the user. Finally, the user unblinds the signing result to obtain the exact signature on the message by performing a decryption-like operation (or an unblinding operation) on the signing result he receives. The signature on the message can be verified by checking whether the corresponding public verification formula with the signature-message pair as parameters is true or not. In a secure blind signature scheme, it is computationally infeasible for the signer to derive the link between a signature and the instance of the signing protocol which produces the blinded form of that signature. This is usually referred to as the unlinkability or blindness property.

In [9], a modified Chaum's blind signature scheme was proposed to enhance the randomization of the signatures against the chosen-text attacks of [6] by injecting randomization factors into the signatures. However, the randomization factors can be removed from the signatures by the users such that the randomization is lost. This manuscript presents a method to repair the weakness and the improved scheme is more user efficient than that of [9].

### 2 Related Works

In this section, we review Chaum's blind signature scheme [2] and Fan-Chen-Yeh blind signature scheme [9].

#### 2.1 Chaum's Blind Signature Scheme

Chaum's blind signature scheme contains five stages, initializing, blinding, signing, unblinding, and verifying. In the initializing stage, the signer publishes the necessary information such as the public keys. The stage can be pre-performed just once. To request the signature on a message, the user blinds the message and submits the blinded message to the signer in the blinding stage. In the signing stage, the signer signs on the blinded message and sends the signing result back to the user. After receiving the signing result, the user performs the unblinding operation on it to obtain the exact signature on the message in the unbinding stage. Finally, the signature is verified in the verifying stage. The protocol is described below.

(1) Initializing. Initially, the signer randomly selects two distinct large primes \( p \) and \( q \), and then computes \( n = pq \) and \( \Phi(n) = (p-1)(q-1) \). The signer chooses two integers \( e \) and \( d \) at random such that \( ed \equiv 1 \pmod{\Phi(n)} \). Then, it publishes \((e, n)\) and a one-way hash function \( H \) such as SHA-1 [13].

(2) Blinding. A user chooses a message \( m \) and randomly selects an integer \( r \) in \( \mathbb{Z}_n^* \) which is the set of all positive integers less than and relatively prime to \( n \). The user computes and submits the integer \( \alpha = (r^e H(m) \mod n) \) to the signer.

(3) Signing. After receiving \( \alpha \), the signer computes and sends the integer \( t = (\alpha^d \mod n) \) to the user.

(4) Unblinding. After receiving \( t \), the user performs the unblinding process to obtain \( s = (r^t \mod n) \). The integer \( s \) is the signature on \( m \).

(5) Verifying. The signature-message pair \((s, m)\) can be verified by checking if \( s^e \equiv H(m) \mod n \).

Given \((s, m)\), the signer cannot derive the link between \((s, m)\) and \( \alpha \) due to the blinding factor \( r \). This is the unlinkability or blindness property.

#### 2.2 Fan-Chen-Yeh Blind Signature Scheme

Fan-Chen-Yeh blind signature scheme [9] is a variant of Chaum's scheme with an injected randomization factor for each issued signature. The scheme is described below.

(1) Initializing. According to the key generation of Chaum's blind signature scheme shown in section 2.1, the public and private keys of the signer are \((e, n)\) and \((p, q, d)\), respectively. Let \( H \) be a public one-
way hash function.

(2) **Blinding.** To request a signature on a message $m$, the user randomly chooses an integer $r$ in $\mathbb{Z}_n^*$ and a positive integer $u$ less than $n$, and then computes and submits the integer $\alpha = (r^2 H(m)(u^2+1) \mod n)$ to the signer. After receiving $\alpha$, the signer randomly selects a positive integer $x$ less than $n$ and sends it to the user. After receiving $x$, the user randomly chooses an integer $b$ in $\mathbb{Z}_n^*$, and then computes $\beta = (b^r(u-x) \mod n)$. Finally, the user submits the integer $\beta$ to the signer.

(3) **Signing.** After receiving $\beta$, the signer computes $t = ((\alpha (x^2+1)\beta^{-2})^e \mod n)$, and then the signer sends $t$ to the user. The integer $x$ is said to be the randomizing factor.

(4) **Unblinding.** After receiving $t$, the user computes

$$
\begin{align*}
  c &= (ux+1)(u-x)^{-1} \mod n \\
  s &= r^{-1}b^{-2}t \mod n.
\end{align*}
$$

(5) **Verifying.** The integer $s$ is the signature on the tuple $(c, m)$. To verify $(c, m, s)$, one can examine if $s^e \equiv H(m)(c^2+1) \mod n$.

However, the randomization factor $x$ can be removed by the user. In the blinding stage, the user forms $\beta = \theta^{(x^2+1)\frac{1}{2}} \mod n$ with an arbitrary odd number $\theta$ such as $\theta = 1$, and sends $\beta$ to the signer. After receiving $\beta$, the signer computes $t = (\alpha (x^2+1)\beta^{-2})^e \equiv (\alpha (x^2+1)(x^2+1)^{2e^{-2}})^e \equiv (x^2+1)^n\alpha^e \mod n$, and sends $t$ to the user. After receiving $t$, the user computes $((x^2+1)^et \mod n)$ to obtain $(\alpha^e \mod n)$ which does not contain the randomization factor $x$.

3 **User Efficient Randomized Chaum’s Blind Signature Scheme**

In this section we present a randomized Chaum’s blind signature scheme to repair the weakness of the randomization in [9] and make it efficient for the users to request and verify the signatures. The details of the proposed scheme are described as follows.

(1) **Initializing.** Initially, the signer randomly selects two distinct large primes $p$ and $q$ such that $p \equiv q \equiv 3 \mod 4$, and then computes $n = pq$ and $\Phi(n) = (p-1)(q-1)$. The signer chooses two integers $e$ and $d$ at random such that $ed \equiv 1 \mod \Phi(n)$. Then, it publishes $(e, n)$ and a one-way hash function $H$.

(2) **Blinding.** To request a signature on a message $m$, the user randomly chooses two integers $r, v$ in $\mathbb{Z}_n^*$ and a positive integer $u$ less than $n$, and then computes and submits the integer $\alpha = (r^2 H(m)(u^2+1) \mod n)$ to the signer. After receiving $\alpha$, the signer randomly selects a positive integer $x$ less than $n$ such that $\alpha(x^2+1) \mod n$ is a quadratic residue (QR) in $\mathbb{Z}_n^*$, and then sends $x$ to the user. After receiving $x$, the user computes $b = (rv \mod n)$, $\Box = (b^r \mod n)$, and $\beta = (\Box (u-x) \mod n)$. Finally, the user submits the integer $\beta$ to the signer.

(3) **Signing.** After receiving $\beta$, the signer computes $\Box = (\beta^e \mod n)$ and a square root $t$ of $((\alpha(x^2+1)\Box^{-2})^e \mod n)$ in $\mathbb{Z}_n^*$ such that $t^2 \equiv (\alpha(x^2+1)\Box^{-2})^e \mod n$.

The signer sends $t$ and $\Box$ to the user. The integer $x$ is the randomizing factor.

(4) **Unblinding.** After receiving $(t, \Box)$, the user computes

$$
\begin{align*}
  c &= \delta t (ux+1) \mod n \\
  s &= tv \mod n
\end{align*}
$$

(5) **Verifying.** The integer $s$ is the signature on the tuple $(c, m)$. To verify $(c, m, s)$, one can examine if $s^{2e} \equiv H(m)(c^2+1) \mod n$.

---

1 Under a modulus $n$, $w$ is a quadratic residue (QR) in $\mathbb{Z}_n^*$ if and only if there exists an integer $y$ in $\mathbb{Z}_n^*$ such that $y^2 \equiv w \mod n$. Given $w$ and $n$, it is intractable to compute the square root $y$ of $w$ in $\mathbb{Z}_n^*$ if $n$ contains large prime factors and the factorization of $n$ is unknown [19].
4 Discussions

In this section we examine the correctness and security of the proposed scheme. First, from the protocol of section 3, we have the following theorem to ensure the correctness of the protocol.

**Theorem 1.** If a triple \((c, m, s)\) is produced by the proposed scheme, then

\[ s \equiv H(m)(c^2 + 1) \pmod{n} \]

Proof. By the Chinese remainder theorem [21], an integer \( w \) in \( Z^*_n \) can be represented by \(< w_1, w_2 >\) where \( w_i = (w \mod p_i) \) and \( w_2 = (w \mod p_2) \). For convenience, \(< w_1, w_2 >\) is denoted by \(< w >\) sometimes. For each \( k > 0 \), let \( k \equiv H(m)(c^2 + 1) \pmod{n} \) be one of the square roots of \( (H(m)(c^2 + 1)) \pmod{n} \) in \( Z^*_n \) [19]. Let \( \equiv H(m)(c^2 + 1) \pmod{n} \) be one of the square roots of the integer \( (H(m)(c^2 + 1)) \pmod{n} \) in \( Z^*_n \), then the four square roots of the integer in \( Z^*_n \) are \(< \pm w_1 \mod p_1, \pm w_2 \mod p_2 >\). Thus, the four square roots of \( b^2 + r^2 \equiv H(m)(c^2 + 1) \pmod{n} \) in \( Z^*_n \) are \(< \pm b_1 r_1 w_1 \mod p_1, \pm b_2 r_2 w_2 \mod p_2 >\). As \( t \equiv b^2 + r^2 \equiv H(m)(c^2 + 1) \pmod{n} \), \( t \) belongs to \(< \pm b_1 r_1 w_1 \mod p_1, \pm b_2 r_2 w_2 \mod p_2 >\). Thus, \( \equiv b^2 + r^2 \equiv H(m)(c^2 + 1) \pmod{n} \) is a QR in \( Z^*_n \), too. Because

\[
\left[ \frac{b^2 + r^2}{p_1} \right] = \left[ \frac{b^2 + r^2}{p_2} \right] = 1
\]

the integer \( (H(m)(c^2 + 1)) \pmod{n} \) also is a QR in \( Z^*_n \) and there are 4 different square roots of \( (H(m)(c^2 + 1)) \pmod{n} \) in \( Z^*_n \) [19]. Let \( \equiv H(m)(c^2 + 1) \pmod{n} \) be one of the square roots of the integer \( (H(m)(c^2 + 1)) \pmod{n} \) in \( Z^*_n \), then the four square roots of the integer in \( Z^*_n \) are \(< \pm w_1 \mod p_1, \pm w_2 \mod p_2 >\).

4.1 Randomization

In the proposed scheme, the attackers can choose \( m \) but that they cannot choose \((c, m)\) on which a signature is computed due to the randomizing factor. 

In the blinding stage, if the user forms \(\beta = \frac{w x}{(x^2 + 1)^2} \pmod{n} \) with an odd number \(\theta\), and sends \(\theta\) to the signer. After receiving \(\theta\), the signer computes an integer \( t \) such that \( t^2 \equiv (\alpha x^2 + 1) \pmod{n} \), and sends \( t \) to the user. After receiving \( t \), the user cannot derive any of the four square roots of \((x^2 + 1) \pmod{n} \) to remove \( x \) from \( t \) since \(\theta\) is odd and computing a square root of the integer in \( Z^*_n \) is intractable without the factorization of \( n \) [19].

Given an integer \( s \), attackers can derive \((u, y)\) such that \( s^2 \equiv (u^2 + y^2) \pmod{n} \) by [16] without \( p \) and \( q \). However, it is still intractable to compute a square root \( c \) of \((w^2 + y^2 - 1) \pmod{n} \) such that \( c^2 \equiv (c^2 + 1) \pmod{n} \) without
the factorization of $n$ \[19\], and deriving an integer $s$ such that $(c')^{s} \equiv ((y^i \omega)^{2} + 1) \pmod{n}$ depends on the security of \[20\] since $s' = (y^{-i} \cdot s \mod{n})$.

### 4.2 Unlinkability

For each instance, numbered $i$, of the proposed protocol, the signer can record the transmitted messages $(\alpha, \beta, x_i)$ between the user and the signer during the instance $i$ of the protocol. The triple $(\alpha, \beta, x_i)$ is usually referred to as the view of the signer to the instance $i$ of the protocol. Thus, we have the following theorem.

**Theorem 2.** Given a triple $(c, m, s)$ produced by the proposed scheme, the signer can derive $b'_i$, $r'_i$, and $u'_i$ for each $(\alpha, \beta, x_i)$ such that:

$$
c \equiv (u'_i \cdot x_i + 1) (u'_i \cdot x_i - 1)^{-1} \pmod{n},$$

$$\alpha_i \equiv (r'_i)^2 H(m)(u'_i)^2 + 1) \pmod{n},$$

$$\beta_i \equiv (b'_i)^2 (u'_i - x_i) \pmod{n}.$$  

**Proof.** If $c \equiv (u'_i \cdot x_i + 1)(u'_i \cdot x_i - 1)^{-1} \pmod{n}$, we have that $u'_i \equiv (c x_i + 1)(c - x_i)^{-1} \pmod{n}$.

If $\alpha_i \equiv (r'_i)^2 H(m)(u'_i)^2 + 1) \pmod{n}$, then we have the followings,

$$\alpha' \equiv (r'_i)^2 H(m)(c x_i + 1)^2 + (c - x_i)^2 \pmod{n},$$

$$\alpha' \equiv (r'_i)^2 H(m)(c x_i + 1)^2 + (c - x_i)^2 \pmod{n},$$

$$\alpha' \equiv (r'_i)^2 H(m)(c x_i + 1)^2 + (c - x_i)^2 \pmod{n},$$

$$\alpha' \equiv (r'_i)^2 H(m)(c x_i + 1)^2 + (c - x_i)^2 \pmod{n},$$

$$\alpha' \equiv (r'_i)^2 H(m)(c x_i + 1)^2 + (c - x_i)^2 \pmod{n},$$

Since $(\alpha'(c x_i + 1)^2) \equiv 0 \pmod{n}$, $(s')^2 \equiv 0 \pmod{n}$, and $((c - x_i)^2 \pmod{n})$ are QR's, the signer can obtain 4 different values of $r'_i$ in $Z^n$.

If $\beta_i \equiv (b'_i)^2 (u'_i \cdot x_i) \pmod{n}$, we have that

$$\beta_i \equiv (b'_i)^2 ((c x_i + 1)(c - x_i)^2 - x_i) \pmod{n},$$

$$(b'_i)^2 \equiv \beta_i ((c x_i + 1)(c - x_i)^2 - x_i) \pmod{n},$$

$$(b'_i)^2 \equiv \beta_i ((c x_i + 1)(c - x_i)^2 - x_i) \pmod{n}.\]

Hence, given a triple $(c, m, s)$ produced by the protocol, the signer can always derive the three blinding factors $b'_i$, $r'_i$, and $u'_i$ for each view $(\alpha, \beta, x_i)$. It turns out that all of the signature-message triples are indistinguishable from the signer's point of view. Therefore, it is computationally infeasible for the signer to derive the link between an instance $i$ of the protocol and the blind signature produced by that protocol.

### 4.3 Performance

Typically, under a modulus $n$, the computation time for a modular exponentiation operation is about $O(n)$ times that of a modular multiplication where $n$ denotes the bit length of $n$ \[21\]. The modulus $n$ is usually taken about 1024 bits or more in a practical implementation \[13, 21\]. In \[4, 8\], some fast modular exponentiation algorithms are proposed. In \[8\], it requires $0.3381|n|$ modular multiplications and large amount of storage, e.g. 83370 stored values for a 512-bit modulus, to perform a modular exponentiation computation. An enhanced version of \[8\] is introduced in \[4\]. However, it still requires $0.3246|n|$ modular multiplications and large amount of storage, e.g. 36027 stored values for a 512-bit modulus, to perform a modular exponentiation computation \[4\]. Besides, an inverse computation in $Z^n$ takes about the same time as that of a modular exponentiation computation in $Z^n$, and a hashing computation does not take longer time than that of a modular multiplication computation \[21\].

In the proposed blind signature scheme, 3 modular exponentiation computations are performed by a user, while 3 modular exponentiations and 2 inverse computations are required for a user in the original scheme \[9\]. Compared to \[9\], the proposed scheme reduces the amount of computations for users by about 40%. In addition, if a short public key $e = 3$ is adopted and we take a modular exponentiation computation to be $0.3246|n|$ modular multiplications \[4\], the proposed method largely reduces the amount of computations for
users by more than 95% under a 1024-bit modulus since no inverse computation and modular exponentiation is needed for users in the proposed scheme.

In the proposed scheme, the signer performs 1 modular exponentiation computation, 1 square root computation, and 1 inverse computation in $\mathbb{Z}_n^*$. Comparing with [9], the proposed protocol does not decrease the computation load for the signer. However, in most of the applications based on blind signatures, the signer usually possesses much more computation capabilities than a user such as the bank of an untraceable electronic cash system or the tally center of an anonymous electronic voting protocol, while the computation capabilities of the users are limited in some situations such as mobile clients and smart-card users. Hence, to guarantee the quality of these ever-growing popular communication services based on blind signatures, it is more urgent to reduce the computation load for the users than that for the signer.

5 Conclusions
We have proposed an improved randomized Chaum's blind signature scheme. The scheme greatly reduces users' computations for mobile and smart-card environments. By performing an additional square-root operation for signing, the weakness of the randomization in the original scheme has been repaired.

References


