A New Blind ECDSA Scheme for Bitcoin Transaction Anonymity

Xun Yi [1,2], Kwok-Yan Lam [2] and Dieter Gollmann [2,3]

[1]RMIT University, Australia, E-mail: xun.yi@rmit.edu.au
[2]Nanyang Technological University, Singapore, E-mail: kwokyan.lam@ntu.edu.sg
[3]Hamburg University of Technology, Germany, E-mail: diego@tu-harburg.de

Abstract. In this paper, we consider a scenario where a bitcoin liquidity provider sells bitcoins to clients. When a client pays for a bitcoin online, the provider is able to link the client's payment information to the bitcoin sold to that client. To address the clients' privacy concern, it is desirable for the provider to perform the bitcoin transaction with blind signatures. However, existing blind signature schemes are incompatible with the Elliptic Curve Digital Signature Algorithm (ECDSA) which is used by most of the existing bitcoin protocol, thus cannot be applied directly in Bitcoin. In this paper, we propose a new blind signature scheme that allows generating a blind signature compatible with the standard ECDSA. Afterwards, we make use of the new scheme to achieve bitcoin transaction anonymity. The new scheme is built on a variant of the Paillier cryptosystem and its homomorphic properties. As long as the modified Paillier cryptosystem is semantically secure, the new blind signature scheme has blindness and unforgeability.

Keywords: Blind signature, ECDSA, Paillier cryptosystem, Bitcoin, Blockchain

1 Introduction

Bitcoin is a peer-to-peer payment system and digital currency introduced as open source software by pseudonymous developer Satoshi Nakamoto [14]. In January 2009, the bitcoin network came into existence with the release of the first bitcoin client and the issuance of the first bitcoins. Bitcoin is a cryptocurrency, so-called because it uses cryptography to control the creation and transfer of money. Bitcoin is the first decentralized digital currency, as the system works without a central bank or single administrator. In this paper, we use the term "Bitcoin" to refer to the technology and "bitcoin" to denote the currency unit of the cryptocurrency.

In Bitcoin, the transactions take place between users directly, without an intermediary. Users send payments, such as payer A sends m bitcoins to payee

B, by broadcasting digitally signed messages to a peer-to-peer network of communicating nodes. Participating nodes, known as miners, verify, timestamp and group newly broadcasted transactions into a new block of the chain, which is then broadcasted to the network and verified by payee nodes. Currently, Elliptic Curve Digital Signature Algorithm (ECDSA) is used by Bitcoin implementations.

An increasing number of online merchants now offer the option to pay using bitcoins. One of the great promises of Bitcoin is anonymity: the transactions are recorded and made public, but they are linked only with an electronic address instead of real-world identity. Hence, whatever you buy with your bitcoins, the purchase cannot be traced specifically to you.

Generally speaking, the aim of anonymization is to prevent attackers from discovering the relationship between bitcoin wallet addresses and the real user identity information through the Bitcoin network and the blockchain. The anonymity feature of bitcoin transactions is by no means perfect.

Let us consider a scenario where a bitcoin liquidity provider sells bitcoins to its clients. If a client pays to the provider for the bitcoin online, e.g., through bank transfer, credit card, PayPal, or even Ali pay, the provider is able to link the client's payment information to the electronic address of the transacted bitcoin. To address this privacy concern of the clients, it is desirable for the bitcoin transactions to be performed by using blind signatures.

Blind signature, as introduced by Chaum [3], a form of digital signature in which the content of a message is disguised (blinded) before it is signed. The resulting blind signature can be publicly verified against the original, unblinded message in the manner of a regular digital signature. A typical application of blind signature is digital cash.

One of the key requirements for digital cash is anonymity: when you take money out of the bank, the bank gives you the cash without knowing what you intend to buy, and when you spend money, the merchant has no idea who you are. In contrast, when you buy something with a credit card online, you have to tell the merchant who you are, and also you have to tell the payment service provider who you are making a purchase from. The potential for intrusion of privacy is immense.

For the purposes of this construction, let us assume that all coins are worth a dollar in real-world currency denomination. To withdraw a dollar from her account, Alice generates a coin C, applies a public hash function H, and masks the result by encrypting it with E_a. The bank signs $E_a(H(C))$ with S_b and debits Alice's bank account. Alice then computes $D_a(S_b(E_a(H(C))))$ to strip away her encryption, leaving her with a signature $S_b(H(C))$, and checks to make sure $V_b(S_b(H(C))) = H(C)$. To spend her coin, Alice then gives the signature $S_b(H(C))$ and C to a merchant. The merchant then computes $V_b(S_b(H(C)))$ and compares that to $H(C)$ in order to make sure that the coin was actually signed by the bank. Then the merchant sends $S_b(H(C))$ and C to the bank, which checks the validity of the signature, transfer fund to the merchant and puts C on a list of coins that have already been spent.

This scheme preserves Alice's anonymity because blind signature prevents the bank from linking the blinded message $E_a(H(C))$ it signs to a later un-blinded version C that it may be called upon to verify.

The same kind of idea can be used to address the privacy concern of clients when a bitcoin provider sells bitcoins to clients. The privacy requirement is that the bitcoin provider cannot tell the relationship between the real identities of clients and the transacted bitcoins in the blockchain. For example, assume that the provider has sold bitcoins to n different clients (where $n \geq 2$) who have broadcast the transacted bitcoins in the blockchain. We require that the success probability for the provider to guess the real identity of a transacted bitcoin from the provider is not more than $1/n$.

Our Contribution. In Bitcoin, the Elliptic Curve Digital Signature Algorithm (ECDSA) is used to verify bitcoin transactions[1]. ECDSA offers a variant of the Digital Signature Algorithm (DSA) [5] using the elliptic curve cryptography. Existing blind signature schemes lack compatibility with the standard ECDSA and thus cannot be used directly. This is mainly because existing blind signature schemes require different signature verification from that in the standard ECDSA. In this paper, we propose a new blind signature scheme that allows generating a blind signature which is compatible with the standard ECDSA in the Bitcoin protocol. Afterwards, we make use of the new scheme to protect the privacy of clients in Bitcoin.

The basic idea is: when a client buys a bitcoin from the bitcoin provider, he pays the provider online at first and then runs a blind signature scheme with the provider to obtain a blind ECDSA signature of the provider on a bitcoin transaction in the same way as digital cash. The bitcoin transaction is created by the client so that the provider does not know the address of the client in the transaction when the provider signs the transaction. Later, the client broadcasts the transaction to the network. The blind signature in the transaction can be verified by everyone with the ECDSA. Since the blind signature is used, the bitcoin provider cannot tell the relationship between the real identities of clients and the transacted bitcoins in the blockchain.

The new blind signature is based on a variant of the Paillier cryptosystem. We formally prove that the modified Paillier cryptosystem has semantic security. Due to this, the blind signature scheme has blindness and unforgeability.

2 Related Work

In 1982, Chaum [3] gave the first blind signature scheme based on RSA [18] and later used the blind signature scheme to construct the first electronic cash scheme in [4].

Assume that a user, called a recipient, wishes a signer to sign a message m without knowing the content of the message and the RSA public and private key pair of the signer is $((n, e), d)$, where n is the product of two large distinct

[1] https://bitcoin.org/en/glossary/signature

primes p and q and $ed = 1(mod\ (p-1)(q-1))$. Chaum's blind signature scheme [3] can be described as follows:

- Step 1. The recipient randomly chooses an integer r from $\mathbb{Z}_n^*$ and computes $m' = r^e m\ (mod\ n)$ and sends the blinded message m' to the signer.
- Step 2. Like the RSA signature scheme, the signer signs the blinded message m' with his private key d by computing $s' = m'^d\ (mod\ n)$ and returns s' to the recipient.
- Step 3. The recipient unblinds the signed message to get the signature of the signer on the message by computing

$$s = r^{-1}s' = r^{-1}m'^d = r^{-1}(r^e m)^d = r^{-1}r^{ed}m^d = m^d\ (mod\ n)$$

  where $r^{ed} = r(mod\ n)$ according to the Euler's theorem.

In 1989, Schnorr [19] proposed a signature scheme based on the intractability of certain discrete logarithm problems. The Schnorr signature scheme can also be turned into a blind signature scheme [16, 17].

In the Schnorr signature scheme, there are two large primes p and q, such that $q|p-1$, an element g of order q modulo p, and a secure cryptographic hash function $H(\cdot)$. The signer generates a pair of public and private keys (y, x), such that $y = g^x(mod\ p)$, where the signer randomly chooses x from $\mathbb{Z}_q^*$. When a recipient wishes the signer to blindly sign a message m, Schnorr's blind signature scheme can be described as follows:

- Step 1. The signer randomly chooses an integer k from $\mathbb{Z}_q^*$ and computes $r = g^k\ (mod\ p)$ and sends the commitment r to the recipient.
- Step 2. The recipient randomly chooses two integers α, β from $\mathbb{Z}_q^*$ and computes $r' = rg^{-\alpha}y^{-\beta}(mod\ p)$, $e' = H(m, r')$ and $e = e' + \beta(mod\ q)$. Then the recipient sends e to the signer.
- Step 3. The signer computes $s = k - ex(mod\ q)$ and sends s to the recipient.
- Step 4 The recipient computes $s' = s - \alpha$. In the end, the recipient obtains a valid Schnorr signature (e', s'), such that $e' = H(m, g^{s'}y^{e'}(mod\ p))$.

Bitcoin uses the Elliptic Curve Digital Signature Algorithm (ECDSA) to verify transactions. The above blind signature schemes are not based on elliptic curve cryptography and therefore cannot be used in Bitcoin. The ECDSA is composed of four algorithms as follows.

- Parameter generation: The algorithm chooses an elliptic curve $\mathbb{E}$ and a group generator $\mathcal{G}$ of prime order q over the elliptic curve. In addition, the algorithm selects a cryptographic hash function $H(\cdot)$. The algorithm parameters $(\mathbb{E}, \mathcal{G}, q, H)$ are shared between different users.
- Key generation: The signer randomly chooses his private key sk from 2 to $q-1$ and computes and publishes his public key $PK = sk\mathcal{G}$.
- Signature generation: To sign a message M, the signer randomly chooses an integer k from 2 to $q-1$ and computes $(K_x, K_y) = k\mathcal{G}$, $s = k^{-1}(H(M) + k_x sk)(mod\ q)$. The signature of the signer on the message M is (K_x, s).

– Signature verification: Anyone has the public key PK of the signer can verify the signature (K_x, s) by computing $u = s^{-1}H(M)(mod\, q), v = s^{-1}K_x(mod\, q)$, $(K'_x, K'_y) = u\mathcal{G} + vPK$ and checking if $K'_x = K_x$. If so, the signature is valid.

If the signature is generated with the signature generation algorithm, we have $(K'_x, K'_y) = u\mathcal{G} + vPK = s^{-1}H(M)\mathcal{G} + s^{-1}K_x sk\mathcal{G} = (k^{-1}(H(M) + K_x sk))^{-1}(H(M) + K_x sk)\mathcal{G} = k\mathcal{G} = (K_x, K_y)$. Therefore, we have $K'_x = K_x$.

In 2015, ShenTu and Yu [20] proposed a blind signature scheme for Bitcoin based on elliptic curve cryptography. Their basic idea is the same as the Schnorr signature scheme [19]. Assume that the scheme chooses an elliptic curve $\mathbb{E}$, a group generator $\mathcal{G}$ over $\mathbb{E}$ with a prime order q, and a cryptographic hash function $H(\cdot)$. The public and private keys of the signer are (Y, x), where $Y = x\mathcal{G}$. The scheme can be described as follows.

– Step 1. The signer randomly chooses an integer k from $\mathbb{Z}_q^*$ and computes $R = k\mathcal{G}$ and sends the commitment R to the recipient.
– Step 2. The recipient randomly chooses two integers α, β from $\mathbb{Z}_q^*$ and computes $R' = R - \alpha\mathcal{G} - \beta Y$ and takes the x-coordinate of R' and let $t = R'_x$. Then the recipient computes $e' = H(m, t)$ and $e = e' + \beta(mod\, q)$ and sends e to the signer.
– Step 3. The signer computes $s = k - ex(mod\, q)$ and sends s to the recipient.
– Step 4 The recipient computes $s' = s - \alpha(mod\, q)$. In the end, the recipient obtains a valid Schnorr signature (e', s'), such that $s'\mathcal{G} + e'Y = R$ and $H(m, t) = e'$, where $t = R'_x$.

The blind signature scheme over elliptic curve proposed in [20] is not directly applicable in implementing Bitcoin transactions because the proposed blind signature is different from that of ECDSA signature. Before Bitcoin appeared, Metet [11] had given a blind signature scheme compatible with DSA and ECDSA in 2004 on the basis of [12]. The scheme is based on the Paillier cryptosystem [15]. In 2012, Ladd [10] used Metet's scheme for a bitcoin transaction anonymity. The Paillier cryptosystem can be described as follows.

– Key generation: A user randomly choose two large distinct primes p, q and an element g of $\mathbb{Z}_{N^2}^*$ whose order is a nonzero multiple of $N = pq$, publishes the public keys (N, g), and keeps the private keys (p, q) secret.
– Encryption: Given the public key (N, g) of the user, one can encrypt a message m, where m is a positive integer less than N, by randomly choosing r from $\mathbb{Z}_{N^2}^*$ and computing $c = E(m) = g^m r^N\ (mod\ N^2)$, where c is the ciphertext of m. Since r is randomly chosen, the ciphertext c of a message m is random. Therefore, Paillier cryptosystem is a probabilistic encryption.
– Decryption: The user can decrypt the ciphertext c with the private key (p, q) by computing $m = \frac{c^\lambda (mod\ N^2) - 1}{N}(p-1)^{-1}(q-1)^{-1}(mod\ N)$ where $\lambda = lcm(p-1, q-1)$.

Paillier cryptosystem has two homomorphic encryption properties as follows: $E(m_1)E(m_2) = E(m_1 + m_2), E(m)^a = E(am)$ for any $m_1, m_2, m, a \in \mathbb{Z}_N$.

In [11, 10], assume that ECDSA chooses an elliptic curve $\mathbb{E}$ and a group generator $\mathcal{G}$ of prime order q, the public and private keys of the signer are (A, x_a), such that $A = x_a\mathcal{G}$. When the recipient wishes to get the signer's blind signature on a message h (a hash value), the blind signature scheme can be described as follows.

- Step 1. The recipient begins by picking a Paillier cryptosystem with a public key n, whose size is between q^6 and q^7, and sends n to the signer.
- Step 2. The signer randomly chooses an integer k_a in the range $[1, q-1]$ and sends $k_a\mathcal{G}$ to the recipient.
- Step 3. The recipient randomly chooses an integer k_b and computes $k_b(k_a\mathcal{G})$. Let t be the x-coordinate of that point, and let $z_a = 1/k_a$ and $z_b = 1/k_b$. Then the recipient sends Paillier encryptions $c_1 = E(tz_b(mod\ q))$ and $c_2 = E(hz_b(mod\ q))$ along with a proof as in [2] that c_1 and c_2 are encryptions of integers less than q and greater than 1.
- Step 4. The Paillier cryptosystem is additively homomorphic and permits efficient multiplication of plaintexts by constants. Multiplying ciphertexts is addition of the plaintext values, and exponentiation is multiplication by a constant. Based on the properties, the signer computes $c = c_1^{x_a z_a} c_2^{z_a} E(rq)$, where r is a random integer in the range $[1, q^5]$, and returns c to the recipient.
- Step 5. The recipient decrypts c to obtain s, the other half of the signature, after taking it modulo q. As the public key is large enough to prevent overflow, this gives the correct answer.

This scheme is compatible with the ECDSA in the bitcoin protocol, but the proof that c_1 and c_2 are Paillier encryptions of integers less than q and greater than 1 has not been provided. In addition, the recipient can obtain not only $s = (k_a k_b)^{-1}(h + tx_a)(mod\ q)$ but also $k_a^{-1}(mod\ q)(k_b^{-1}H)(mod\ q) + k_a^{-1}(mod\ q)(k_b^{-1}t(mod\ q))x_a + dq$. It is unclear if this has any impact on the security of the signature scheme.

3 New Blind Elliptic Curve Digital Signature Algorithm (ECDSA)

3.1 Modified Paillier Cryptosystem

The proposed blind ECDSA will be built on a variant of the Paillier cryptosystem. In this section, we describe the modified Paillier cryptosystem.

Given a large prime q (a public parameter of ECDSA), the modified Paillier cryptosystem is described as follows.

- Key Generation: A user randomly choose two large distinct primes p, t, such that $gcd(p-1, q) = 1$ and $gcd(t-1, q) = 1$, and computes

$$N = pqt \tag{1}$$
$$g = (1+N)^{pt}(mod\ N^2) \tag{2}$$

  and publishes the public keys (N, g), and keeps the private keys (p, t) secret.

– Encryption: Given the public key (N, g) of the user, one can encrypt a message m, where m is a positive integer less than q, by randomly choosing r from $\mathbb{Z}^*_{N^2}$ and computing

$$C = g^m r^N \ (mod\ N^2) \tag{3}$$

C is the ciphertext of m. Since r is randomly chosen, the ciphertext C of a message m looks like a random number.

– Decryption: The user can decrypt the ciphertext C with the private key (p, t) by computing

$$D = C^{(p-1)(q-1)(t-1)} (mod\ N^2) \tag{4}$$

$$m' = [(D-1)/(Npt)][(p-1)(q-1)(t-1)]^{-1} (mod\ q) \tag{5}$$

Like the original Paillier cryptosystem, the modified scheme also has the homomorphic encryption properties.

Next, we show that the decryption algorithm can recover a ciphertext produced by the encryption algorithm to a plaintext.

Theorem 1 (Correctness). If C is computed as above, we have $m' = m$.

Proof. Since $g = (1+N)^{pt} (mod\ N^2)$, we have

$$g^q = (1+N)^{pqt} = (1+N)^N = 1 (mod\ N^2).$$

Therefore,

$$\begin{aligned} D &= C^{(p-1)(q-1)(t-1)} \\ &= g^{m(p-1)(q-1)(t-1)(mod\ q)} r^{N(p-1)(q-1)(t-1)} \\ &= (1+N)^{pt[m(p-1)(q-1)(t-1)(mod\ q)]} (mod\ N^2) \\ &= 1 + pt[m(p-1)(q-1)(t-1)(mod\ q)]N \end{aligned}$$

where $pt[m(p-1)(q-1)(t-1)(mod\ q)] < N$. Note that according to the Euler theorem, $X^{\phi(N^2)} = 1 (mod\ N^2)$ for any non-zero integer X, where $\phi(N^2) = N(p-1)(q-1)(t-1)$.

In view of this, we have

$$pt[m(p-1)(q-1)(t-1)(mod\ q)] = (D-1)/N.$$

Therefore, we have

$$m(p-1)(q-1)(t-1)(mod\ q) = (D-1)/(Npt),$$

from which we can obtain

$$m = [(D-1)/(Npt)][(p-1)(q-1)(t-1)]^{-1} (mod\ q).$$

Based on Eq. (5), we can see $m' = m$. Therefore, the theorem is true. $\triangle$

3.2 New Blind ECDSA

Based on the modified Paillier cryptosystem described in the above section, we propose a blind ECDSA.

Suppose that the group generator $\mathcal{G}$ of the elliptic curve used by the elliptic curve digital signature algorithm (ECDSA) has a large prime order q. Assume that the recipient wishes the signer (with the public key $PK = sk\mathcal{G}$) to produce a blind signature on the hash value h of his message (e.g., his public key for signature verification), the blind ECDSA between the recipient $\mathcal{R}$ and the signer $\mathcal{S}$ can be described as follows:

- Step 1: The signer $\mathcal{S}$ randomly chooses an integer k_1 from 2 to $q-1$ and computes

$$K_1 = k_1\mathcal{G} \tag{6}$$

  and sends it to the recipient $\mathcal{R}$.
- Step 2: After receiving K_1 from the signer, the recipient $\mathcal{R}$ randomly chooses an integer k_2 from 2 to $q-1$ and computes

$$K = (K_x, K_y) = k_2K_1 \tag{7}$$

  Next, the recipient $\mathcal{R}$ follows the modified Paillier cryptosystem, described in the last session, to chooses two distinct large primes p, t, and computes the public key (N, g). Next, he encrypts h and K_x, respectively, i.e., he randomly chooses r_1, r_2 from $\mathbb{Z}^*_{N^2}$ and computes

$$C_1 = g^h r_1^N (mod\ N^2) \tag{8}$$
$$C_2 = g^{K_x} r_2^N (mod\ N^2) \tag{9}$$

  and submits (N, g, C_1, C_2) to the signer $\mathcal{S}$.
  In addition, the recipient $\mathcal{R}$ needs to prove to the signer that the encryptions are constructed correctly by zero-knowledge proof. The interactive zero-knowledge proof is described as follows: To prove a ciphertext $C = g^m r^N (mod\ N^2)$ to be constructed according to the scheme, the recipient $\mathcal{R}$ randomly chooses $m' \in \mathbb{Z}_q$ and $r' \in \mathbb{Z}^*_{N^2}$, computes $C' = g^{m'} r'^N (mod\ N^2)$ and submits C' to the signer $\mathcal{S}$, who randomly selects a bit $b \in \{0, 1\}$ and returns b to the recipient $\mathcal{R}$. If $b = 0$, the recipient $\mathcal{R}$ is required to submit m', r' to the signer $\mathcal{S}$, who checks if $C' = g^{m'} r'^N (mod\ N^2)$. If $b = 1$, the recipient $\mathcal{R}$ is required to submit $m'' = m + m' (mod\ q)$ and $r'' = rr' (mod\ N^2)$ to the signer, who checks if $CC' = g^{m''} r''^N (mod\ N^2)$. The above process is repeated for ℓ times between the recipient $\mathcal{R}$ and the signer $\mathcal{S}$. If the recipient $\mathcal{R}$ answers all questions correctly, the signer S is sure that the ciphertext C is constructed correctly with a probability $1 - 1/2^\ell$. The interactive zero-knowledge proof can be transferred to a non-interactive protocol according to [6].

– Step 3: After receiving (N, g, C_1, C_2) and verifying that (C_1, C_2) are constructed correctly, the signer $\mathcal{S}$ randomly choose r from 2 to N and computes

$$C = (C_1 C_2^{sk})^{k_1^{-1}(mod\ q)} r^N (mod\ N^2) \tag{10}$$

and sends C to the recipient $\mathcal{R}$.

– Step 4: After receiving C, the recipient $\mathcal{R}$ computes

$$s = k_2^{-1} D(C, (p, t))(mod\ q) \tag{11}$$

where $D(C, (p, t))$ stands for the decryption algorithm. In the end, the recipient $\mathcal{R}$ obtains a blind signature (K_x, s).

Theorem 2 (Correctness). If K_1, K, C_1, C_2, C, s are computed as above, (K_x, s) is a valid ECDSA signature of the signer.

Proof. Because $C_1 = g^h r_1^N (mod\ N^2), C_2 = g^{K_x} r_2^N (mod\ N^2)$, we have

$$\begin{aligned} C &= (C_1 C_2^{sk})^{k_1^{-1}(mod\ q)} r^N \\ &= (g^h r_1^N (g^{K_x} r_2^N)^{sk})^{k_1^{-1}} r^N \\ &= g^{k_1^{-1}(h + K_x sk)(mod\ q)} (r_1^{k_1^{-1}} r_2^{k_1^{-1} sk} r)^N (mod\ N^2) \end{aligned}$$

Note that $g^q = (1+N)^{pqt} = (1+N)^N = 1(mod\ N^2)$.

Therefore, we have

$$s = k_2^{-1} D(C, (p, t)) = k_2^{-1} k_1^{-1} (h + K_x sk)(mod\ q).$$

In addition,

$$K = (K_x, K_y) = k_2 K_1 = k_2 (k_1 \mathcal{G}) = k_1 k_2 \mathcal{G}$$

According to the ECDSA, (K_x, s) is a valid ECDSA signature. △

4 Anonymous Bitcoin Transaction Based on Blind ECDSA

In this section, we address the anonymity requirement by applying the proposed blind ECDSA to achieve anonymity in Bitcoin transaction. We assume that the bitcoin provider $\mathcal{B}$ has a number of bitcoins with the same public key PK_B, where $PK_B = sk_B \mathcal{G}$ and sk_B is the corresponding signing key of the provider $\mathcal{B}$. When a client $\mathcal{C}$ buys a bitcoin from the provider $\mathcal{B}$, the client $\mathcal{C}$ and the provider $\mathcal{B}$ interact as follows:

– Step 1. The client $\mathcal{C}$ pays the provider $\mathcal{B}$ the amount of a bitcoin by bank transfer, credit card payment, or some other payment methods. Usually, the client $\mathcal{C}$ is required to pay slightly more than the amount of a bitcoin to the provider $\mathcal{B}$, e.g., an extra exchange fee.

- Step 2. The client $\mathcal{C}$ generates his ECDSA public and private key pair (Pk_C, sk_C), such that $PK_C = sk_c\mathcal{G}$, which will be used for next round of transaction, and computes the hash value of his public key, denoted as $h = H(PK_C)$.
- Step 3. The client $\mathcal{C}$ (as the recipient) and the provider $\mathcal{B}$ (as the signer) run the blind ECDSA as described in the last section. In the end, the client $\mathcal{C}$ obtains a blind ECDSA signature (K_x, s) of the provider $\mathcal{B}$ on h.
- Step 4. Later, the client $\mathcal{C}$ broadcasts $\{PK_c, (K_x, s)\}$ in the network of bitcoin nodes and miners. Each node or miner is able to verify the signature of the provider $\mathcal{B}$ on the public key PK_C of the client $\mathcal{C}$ on the basis of the public key PK_B of the provider. Note that one signature of the provider $\mathcal{B}$ implies to transfer only one bitcoin to a client $\mathcal{C}$ from the provider $\mathcal{B}$.
- Step 5. According to the public key PK_C of the client $\mathcal{C}$, a transaction is constructed, such as the provider $\mathcal{B}$ transfers one bitcoin to the client $\mathcal{C}$. After the transaction, the number of bitcoins of the provider $\mathcal{B}$ is reduced by one, and the client $\mathcal{C}$ has one bitcoin.

Remark. When the client $\mathcal{C}$ pays the provider $\mathcal{B}$ the amount of a bitcoin in the first step, the provider $\mathcal{B}$ may know the identity of the client $\mathcal{C}$. However, when the client $\mathcal{C}$ broadcasts the transaction in Step 4, the provider $\mathcal{B}$ can only know that the client $\mathcal{C}$ is one of those clients who bought bitcoins from the provider, but the provider $\mathcal{B}$ cannot tell whom he is.

5 Security Analysis

First of all, we analyze the semantic security of the modified Paillier cryptosystem. We begin by briefly introducing composite degree residues as a natural instance of higher degree residues, and give some basic related facts. The originality of our setting resides in using of a square number as modulus. The modulus is the product of three distinct primes, i.e, $N = pqt$.

Definition 1. An integer z is said to be a N-th residue modulo N^2 if there is an integer y such that $z = y^N (mod\ N^2)$.

The set of N-th residues is a multiplicative subgroup of $\mathbb{Z}^*_{N^2}$ of order $\phi(N) = (p-1)(q-1)(t-1)$. Each N-th residue z has exactly N roots of degree N, among which exactly one is strictly smaller than N. The N-th roots of unity are the numbers of the form $(1+N)^x = 1 + xN (mod\ N^2)$.

The problem of deciding N-th residuosity, i.e. distinguishing N-th residues from non N-th residues will be denoted by $CR[N]$.

As for prime residuosity [1, 13], deciding p-th residuosity is believed to be computationally hard. Accordingly, we assume that:

Conjecture. There exists no polynomial time distinguisher for N-th residues modulo N^2, i.e. $CR[N]$ is intractable.

This intractability hypothesis is referred to as the Decisional Composite Residuosity Assumption (DCRA) throughout this following discussion.

We now proceed to describe the number-theoretic framework underlying the modified Paillier cryptosystem introduced in sections 4.1. Let $g = (1+N)^{pt}$ and denote by $E(m, r)$ the integer-valued function defined by

$$E(m, r) = g^m r^N (mod\ N^2)$$

where $m \in \mathbb{Z}_q$ and $r \in \mathbb{Z}^*_{N^2}$. Note that $g^q = 1(mod\ N^2)$.

Definition 2. We call Composite Residuosity Class Problem the computational problem $Class[N]$ defined as follows: for a given $C = E(m, r) \in \mathbb{Z}^*_{N^2}$ where $m \in \mathbb{Z}_q$ and $r \in \mathbb{Z}^*_{N^2}$, compute m.

Associated to the computation problem, we define the decisional problem as follows:

Definition 3. Associated to $Class[N]$, we call the decisional problem $D\text{-}Class[N]$ defined as: given $C = E(m, r) \in \mathbb{Z}_{N^2}$ and $x \in \mathbb{Z}_q$, decide if $x = m$.

Based on the above definitions, we have

Theorem 3. There exists the following computational hierarchy

$$CR[N] = D\text{-}Class[N] \Leftarrow Class[N]$$

Proof. The hierarchy $D\text{-}Class[N] \Leftarrow Class[N]$ comes from the general fact that it is easier to verify a solution than to compute it.

Let us prove the left-side equivalence.

($\Rightarrow$) Assume that $C = g^m r^N (mod\ N^2)$ and submit $Cg^{-x} = g^{m-x} r^N (mod\ N^2)$ to the oracle solving $CR[N]$. In case of N-th residuosity detection, there exists an integer r' such that $Cg^{-x} = {r'}^N (mod\ N^2)$. Therefore, we have $g^{m-x}(r/r')^N = 1(mod\ N^2)$. Raising two sides to the same power $(p-1)(q-1)(t-1)$, we have $g^{(m-x)(p-1)(q-1)(t-1)(mod\ q)} = 1(mod\ N^2)$, i.e., $1 + pt[(m-x)(p-1)(q-1)(t-1)(mod\ q)]N = 1(mod\ N^2)$. Because $(m-x)(p-1)(q-1)(t-1)(mod\ q) < q$, we have $x = m$. Therefore, the answer is "Yes". Otherwise, the answer is "No".

($\Leftarrow$) Submit the pair $(z = E(m, r), x = 0)$ to the oracle solving $D\text{-}Class[N]$. Return the oracle's answer without change.

Therefore, the theorem is proved. $\triangle$

Remark. In the above proof, we consider some special cases of $CR[N]$, that is, $z = E(m, r) = g^m r^N (mod\ N^2)$ instead of any $z \in \mathbb{Z}_{N^2}$. However, since we assume that $CR[N]$ is intractable, it must be intractable in any case.

Based on the above facts, we now analyze the semantic security of the modified Paillier cryptosystem. The semantic security of a public key crytosystem is commonly defined by the following experiment [7, 8].

- A random public and private pair (pk, sk) are generated by running $\mathsf{Gen}(1^N)$.
- A probabilistic polynomial time-bounded adversary is given the public key pk, which it may use to generate any number of ciphertexts (within polynomial bounds).

– The adversary generates two equal-length messages m_0 and m_1, and transmits them to a challenge oracle along with the public key. The challenge oracle selects one of the messages by flipping a fair coin (selecting a random bit $b \in \{0, 1\}$), encrypts the message m_b under the public key, and returns the resulting challenging ciphertext c to the adversary.

The underlying public key cryptosystem is semantically secure under chosen plaintext attack if the adversary cannot determine which of the two messages was chosen by the oracle, with probability significantly greater than 1/2 (the success rate of random guessing).

Theorem 4. The modified Paillier cryptosystem is semantically secure if $CR[N]$ is intractable.

Proof. Since $CR[N]$ is intractable, $D\text{-}Class[N]$ is also intractable according to Theorem 3. Therefore, given m_0, m_1, and $E(m_b, r)$, the adversary cannot determine which of the two messages was chosen by the oracle, with probability significantly greater than 1/2. The theorem is proved. △

Next, we analyze the security of the proposed blind signature based on the modified Paillier cryptosystem. According to [9], a blind digital signature scheme is secure if it has blindness and unforgeability.

Blindness. Intuitively, blindness of a blind signature means that the recipient $\mathcal{R}$ does not reveal to the signer $\mathcal{S}$ any information about the signature during the generation of the signature.

In the proposed blind signature scheme, the information relevant to the signature include h and (K_x, s). During the signature generation, both h and K_x are encrypted by the recipient $\mathcal{R}$ with the modified Paillier cryptosystem. Based on Theorem 4, the modified Paillier cryptosystem is semantically secure. Therefore, without knowing the decryption key, the signer $\mathcal{S}$ cannot distinguish h, K_x from random integers in $\mathbb{Z}_q$. In addition, although the encryption of $k_1^{-1}(h + K_x sk)(mod\ q)$ is computed by the signer $\mathcal{S}$ based on the homomorphic property of the Paillier cryptosystem, the signer $\mathcal{S}$, without knowing the decryption key, cannot distinguish $s = k_2^{-1}[k_1^{-1}(h + K_x sk)](mod\ q)$ from a random integer in $\mathbb{Z}_q$. If the signer $\mathcal{S}$ returns an encryption of a known integer α instead of $k_1^{-1}(h + K_x sk)(mod\ q)$ to the recipient $\mathcal{R}$, it can be detected by the recipient $\mathcal{R}$ because (K_x, s), where $s = k_2^{-1}\alpha$, is usually not a valid signature of the signer. Even if $(K_x, k_2^{-1}\alpha)$ happens to be a valid signature, the signer $\mathcal{S}$ cannot trace the signature $(K_x, k_2^{-1}\alpha)$ with α because he has no knowledge of k_2 in the signature.

In summary, the recipient $\mathcal{R}$ does not reveal to the signer $\mathcal{S}$ any information about the blind signature. Therefore, the proposed blind signature has blindness.

Unforgeability. Intuitively, unforgeability of a blind signature means that the recipient $\mathcal{R}$ cannot forge any new blind signature if given some blind signatures.

In the proposed blind signature scheme, when the recipient $\mathcal{R}$ submits to the signer $\mathcal{S}$ the encryptions of h and K_x, he is required to provide zero-knowledge proof that the encryptios are constructed correctly. This ensure that $C_1 =$

$g^h r_1^N (mod\ N^2)$ and $C_2 = g^{K_x} r_2^N (mod\ N^2)$. Therefore, $C = (C_1 C_2^{sk})^{k_1^{-1}(mod\ q)} r^N$ $(mod\ N^2) = E(k_1^{-1}(h + K_x sk)(mod\ q))$ from which the recipient $\mathcal{R}$ can only obtain $s = k_2^{-1} k_1^{-1}(h + K_x sk)(mod\ q)$ and then a blind signature (K_x, s) on h. Note that the recipient $\mathcal{R}$ may not obtain a valid signature if he does not follow the scheme to send the encryptions of h and K_x to the signer $\mathcal{S}$.

Since the ECDSA has been assumed to be unforgeable, given some blind ECDSA signatures, the recipient $\mathcal{R}$ cannot forge any new blind ECDSA signature.

In summary, the proposed blind signature scheme has unforgeability if the ECDSA is unforgeable.

6 Performance Analysis

In this section, we analyze the computation and communication complexities required by the new blind signature scheme, where the recipient $\mathcal{R}$ and the signer $\mathcal{S}$ interact to generate the blind signature (K_x, s) on a hash value h.

With reference to Section 4.2, the scheme is described with 4 steps. We will analyze the computation and communication complexities required for the recipient $\mathcal{R}$ and the signer $\mathcal{S}$, respectively, in each step as follows.

Steps	*Recipient Comp.*	*Signer Comp.*	*2-Party Comm. (bits)*
Step 1		1 m_p	$2\|q\|$
Step 2	1 m_p+2 exp.+2ℓ exp.	2ℓ exp.	$(4\ell+2)\|N^2\| + 2\ell\|q\|$
Step 3		3 exp.	1 $\|N^2\|$
Step 4	1 exp.		
Total	1 m_p+$(2\ell+3)$ exp.	1 m_p+$(2\ell+3)$ exp.	$(4\ell+3)\|N^2\|$+$(2\ell+2)\|q\|$

Table 1. Computation and Communication Complexities for Recipient and Signer

In Table 1, m_p represents a point multiplication over an elliptic curve, exp. denotes a modular exponentiation with the modulo N^2, ℓ is the number of interactions between the recipient $\mathcal{R}$ and the signer $\mathcal{S}$ for zero-knowledge proof, and $|x|$ stands for the bit length of an integer x.

Note that in the performance analysis, we have not considered the complexities of computing modular additions and multiplications in the above analysis because they are relatively small compared to the complexities of computing point multiplications and modular exponentiations. In addition, because $g = (1+N)^{pt} (mod\ N^2)$, so $g^m = (1 + mptN)(mod\ N^2)$ and therefore the computation of $C = g^m r^N (mod\ N^2)$ needs only one modular exponentiation instead of two.

From Table 1, it is interesting to see that the computation complexities of the recipient and the signer are the same.

If we run the proposed blind ECDSA ($|p| = 512, |q| = 256, |t| = 512$) on the Skylake Core-i5 test platform with 2.7e+09 Hz CPU frequency, according

to Crypto++ 6.0.0 benchmarks[2], one modular exponentiation with modulo N^2, where $|N^2| = 2560$, takes less than 2 ms and one point multiplication with modulo q, where $|q| = 256$, takes less than 1 ms. In case that $\ell = 20$, the total computation time for the recipient and the signer to generate a blind signature is less than 175 ms, and the total communication overload is less than 28 kbytes.

In the case that a bitcoin provider sells 100 bitcoins to 100 clients in the same time with the proposed blind signature for bitcoin transaction anonymity, the total computation time of the signer is less than $100 \times 175/2 = 8750\ ms = 8.75\ s$.

We can improve the performance of our scheme by using the Chinese Reminder Theorem (CRT) to compute the modular exponentiation $a^x (mod\ N^2)$ where $N = pqt$. The recipient, knowing p, q, t, can compute $a^x (mod\, p^2), a^x (mod\, q^2)$ and $a^x (mod\ t^2)$ at first and then $a^x (mod\ N^2)$ with CRT. The signer, knowing pt and q, can compute $a^x (mod\ p^2 t^2)$ and $a^x (mod\ t^2)$ at first and then $a^x (mod\ N^2)$ with CRT.

We can significantly improve the performance of our scheme by eliminating the zero-knowledge proof. To protect the signing key sk of the signer against the malicious recipient, we can modify Eq. (10) as follows:

$$C = (C_1 C_2^{sk})^{k_1^{-1} (mod\ q) + r' q} r^N (mod\ N^2) \tag{12}$$

where the signer randomly chooses r' from $\mathbb{Z}_q^*$, and the recipient still computes s as Eq. (11). This change brings about the computation complexities of both the recipient and the signer being reduced to only 1 point multiplication plus 3 modular exponentiations and the communication overload being reduced to only $3|N^2| + 2|q|$. Although this change improves the performance significantly, we have not provided a rigorous security proof so far.

At last, let us compare our scheme with the blind ECDSA suggested in [10, 11]. In [10, 11], the size of the modulo n for Pillier cryptosystem is set between q^6 and q^7. When $|q| = 256$, the average size of n is 1664 and therefore $|n^2| = 3328$, which is larger than that in our scheme when $|p| = 512, |q| = 256, |t| = 512$ and $|N^2| = 2560$. In this case, the computation of a modular exponentiation in [10, 11] takes longer time than that in our scheme. In addition, Metet [11] has not provided any security proof of the suggested blind signature scheme.

7 Conclusion

In the bitcoin protocol, it is hard to hide the identity of a client who buys a bitcoin from a bitcoin provider when the client pays the provider by bank transfer, credit card, PayPal, or even Ali pay. In this paper, we proposed a new blind ECDSA scheme to achieve bitcoin transaction anonymity. With the new blind ECDSA scheme, the client can pay the provider by any payment approach to exchange a blind signature of the provider on the transaction and later broadcast it in the bitcoin network. The bitcoin provider cannot tell the

[2] https://www.cryptopp.com/benchmarks.html

relationship between the real identities of clients and the sold bitcoins in the blockchain.

We have provided a rigorous security proof that the proposed blind signature scheme has blindness and unforgeability on the basis of the semantic security of the modified Paillier cryptosystem. Performance analysis has shown that our scheme is feasible and can be used by a bitcoin provider to sell bitcoins to multiple client in the same time.

Our future work is to study if we can remove the zero-knowledge proof to improve the performance of our scheme.

Acknowledgements Part of this work was performed by the second co-author at the Schloss Dagstuhl. The authors would like to thank the organizers of the Dagstuhl Seminar 18152: Blockchain, Smart Contracts and Future Applications for providing a conducive environment for constructive discussions on this topic.

References

1. J. C. Benaloh. *Verifiable Secret-Ballot Elections*. PhD Thesis, Yale University, 1988.
2. F. Boudot. Efficient proofs that a committed number lies in an interval. In EUROCRYPT 2000.
3. D. Chaum. Blind signature for untraceable payment. In CRYPTO 1982.
4. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In CRYPTO 1988.
5. Digital Signature Standard (DSS). Digital Signature Standard (DSS). July 2013. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
6. A. Fiat and A., Shamir. How to prove yourself: practical solutions to identification and signature problems. In CRYPTO 1986, pages 186-194.
7. S. Goldwasser and S. Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. Annual ACM Symposium on Theory of Computing, 1982.
8. S. Goldwasser and S. Micali. Probabilistic encryption. Journal of Computer and System Sciences 28: 270-299, 1984.
9. A. Juels, M. Luby, R. Ostrovsky. Security of blind digital signatures. In . CRYPTO 1997, pages 150-164.
10. W. Ladd. Blind signatures for bitcoin transaction anonymity. https://pdfs.semanticscholar.org/db29/892c3641587ab0d3af7be584b0d61546ae72.pdf
11. S. Metet. Blind signature with das/ecdsa? 2004. http://www.metzdowd.com/pipermail/cryptography/2004-April/006790.html
12. P. MacKenzie and M. K. Reiter. Two-party generation of DSA signatures. in CRYPTO 2001, pages 137–154.
13. D. Naccache and J. Stern. A new public-key cryptosystem based on higher residues. In EUROCRYPT 1998, pages 308-318.
14. S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 24 May 2009. https://bitcoin.org/bitcoin.pdf
15. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In EUROCRYPT 1999, pages 223-238.
16. D. Pointcheval and J. Stern. Provably secure blind signature schemes. In ASIACRYPT 1996.

17. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. Journal of Cryptology, 13(3): 361-396, 2000.
18. R. Rivest, A. Shamir and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM. 21 (2): 120-126, 1978.
19. C. P. Schnorr. Efficient identification and signatures for smart cards. In CRYPTO 1989.
20. Q. ShenTu and J. Yu. A blind-mixing scheme for Bitcoin based on an elliptic curve cryptography blind digital signature algorithm. ATR Defense Science & Technology Lab, 2015.