Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Preserving transaction privacy in bitcoin

Qin Wang [a,b], Bo Qin [c], Jiankun Hu [d], Fu Xiao [e,*]

[a] *School of Electronic and Information Engineering, Beihang University, Beijing, China*
[b] *State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, 100093, China*
[c] *Key Laboratory of Data Engineering and Knowledge Engineering, Ministry of Education, School of Information, Renmin University of China, Beijing, China*
[d] *School of Engineering and Information Technology, University of New South Wales Defence Force Academy Canberra, Australia*
[e] *School of Computer, Nanjing University of Posts and Telecommunications, Nanjing, China*

HIGHLIGHTS

- We propose encrypted bitcoin transactions allowing coins to be merged, split, or spent while simultaneously keeping the value transferred.
- We achieve verifiability of encrypted bitcoin transactions by using Commitment Proof.
- We present transaction privacy enhanced bitcoin protocol compatible with the original protocol while only using well established cryptographic assumptions.

ARTICLE INFO

Article history:
Received 31 December 2016
Received in revised form 30 June 2017
Accepted 14 August 2017
Available online xxxx

Keywords:
Bitcoin
Block chain
Transaction privacy
Zero-knowledge proof

ABSTRACT

As bitcoin has received considerable attentions, a large number of transactions are increasingly published through the Internet. The potential threat to online transactions is not only limited to the anonymity of identities such as *address* and *pseudonyms*, but also other sensitive information shown on scripts like transaction amounts. We propose a framework to hide the amounts by employing the Paillier cryptosystem for encryption and decryption. Due to its homomorphic properties, we create a *dumb account* which can receive the same amounts of bitcoins in each transaction but cannot spend them. This ensures that the hidden amounts are always positive, and keeps the equality between the Input-sum and the Output-sum via the commitment proofs. Our framework achieves delicate anonymity and prevents active and passive attacks, which effectively improves the transaction privacy. Analyses show that our proposal is secure and practical.

1. Introduction

With the continuous expansion of human activity scope and the emergence of the high-frequency transactions, the traditional real currency has gradually developed into credit currency in order to meet the remarkably growing needs of the convenience and security. Commercial transactions in digital and virtual currency becomes widespread with the popularity of the Internet. In 1982, David Chaum, the founder of digital currency proposed a new cryptographic primitive: *Blind Signature* [1], and a digital cash system with anonymity and intractability based on which is regarded as the earliest digital monetary theory. A large amounts of follow-on works have improved many aspects of this theory, but the model remains centralized. The financial institutions including banks, governments and arbitral authorities, process the intermediate links over the traders throughout the whole development of the electronic currency in 1990s. Densely-centralized schemes incurred the high risk of trust and additional cost of transactions. A number of startup companies, such as Peppercoin [2], DigiCash [3], intended to translate theory into practice but eventually failed.

In contrast, launched in 2009, Bitcoin has emerged as the first and most popular and successful cryptographic currencies to date. This brand new electronic cash system is naturally more spectacular than others due to its unforgeability, irreversibility, and pseudonymity. Introduced by Satoshi Nakamoto [4], Bitcoin is based on a distributed structure through a P2P computer network. There is no central trust authority to manage/maintain the system, and the participants can transact with any other nodes free of control or supervision under the authorities. Successfully verified transactions can bring extra Bitcoins, as a reward provided to the Bitcoin miners, motivating to produce the continuously unique valid block chain [5]. In general, the traders, the miners and the

* Corresponding author.
E-mail addresses: wangqin0409@foxmail.com (Q. Wang), bo.qin@ruc.edu.cn (B. Qin), J.Hu@adfa.edu.au (J. Hu), xiaof@njupt.edu.cn (F. Xiao).

http://dx.doi.org/10.1016/j.future.2017.08.026
0167-739X/© 2017 Elsevier B.V. All rights reserved.

decentralized nodes as a trading platform make Bitcoin form a self-organized system.

Deeply rooted in cryptography, the Bitcoin system has incorporated several profound conceptions: (a) *Workload proof mechanism (proof-of-work)*, dating back to the early 1990s [6], relaxes the anonymity property to mere pseudonymity. This work is deployed by numerous related coin-systems including Litecoin [7], Zerocoin [8], Primecoin [9] et al. and practical applications, including the fair lottery [10], and minting micropayments [11] et al. (b) *Byzantine agreement(BA)*, dating back to [12,13], considers that n parties intend to output an agreement in the presence of the disruptive behavior. The BA problem is connected with the double-spending problem of Bitcoin in the underlying idea, by analyzing that the adversary controls more than 1/3 and 1/2 of the computational power [14]. (c) *Public ledger*, dating back to [15], establishes a public database to collect all the transactions to prevent the fraud trades. The entire set of valid transactions in Bitcoin replaced by a hashed address recorded on the *Merkle Tree*, and the whole verified transactions form a blockchain. (d) Cryptographic *hash functions*, dating back to early 1970s [16], maps the arbitrary length input strings to the short fixed length output strings with uniform confusion. The mapping process of the address in Bitcoin is based on SHA-256, which belongs to the series algorithm of SHA, designed by the National Institute of standards and Technology (NIST) [17]. Although the collision has been found by Wang et al. [18], SHA series are still the most widely used hash functions. (e) *Elliptic curve*, secp256k1 ECDSA curve in specific [19], creates the private/public keys in Bitcoin, to improve the security of the whole system.

Due to a profound comprehension of the Bitcoin mechanism, considerable research attentions are attracted in both academia and industrial areas. Literatures, such as [20–22] et al. have identified hidden but important properties, potential attacks, and uncertain future challenges of the system. Except for Proof-of-Work(PoW) described above, a new mechanism called Proof-of-Stake(PoS) has emerged [23]. In the POS mode, such as Bitshares [24], interest is distributed according to the time and the amounts of your holdings, and your mining income is proportional to your processed coins. In the meanwhile, a large and vibrant open-source community, such as Ethereum [25], has explored and deployed substantial extensions and applications. The most competitive application of blockchain, also the underlying technology of Bitcoin, is the Smart Contracts. Portending Bitcoin's scripting capabilities, the smart contracts enable parties to specify a cryptographically enforceable agreement automatically [26]. The users can freely encode arbitrary state transition functions to create any of the systems based on it.

Although Bitcoin has made a large number of transactions among different individuals and parties, the potential risk has been ignored. When these transactions begin to flow throughout the whole internet, they face a vulnerability which leads to the leakage of the transaction amounts and threatens the privacy of the traders. The anonymity and intractability of the Bitcoin only cut the links between the real identities and the transaction scripts. It still exposes the amount money in each transaction. Technically, the Bitcoin system addresses the privacy problem by hiding the addresses of transactions via Hash function. But the hash value is obviously unique, if the addresses and the traders are bounded unconsciously through daily activities by attackers, the anonymity of the whole system will disappear. On the one hand, without adequate protection, attackers and scammers can spear their efforts to some special transaction scripts with conspicuously high-value targets, which may leak the real identities of traders and the business details. On the other hand, the incomplete privacy on transactions will lead to the fungibility of the scripts, which could undermine the stability of Bitcoin as a long-lived currency system. Let us consider the following scenario. Suppose that the Bitcoin system is connected to the payment system of e-commerce platform. The strong computing power can search the complete data of your incomes and outcomes, which makes everything about money around you be exposed to the public. More simply, your boss paid you salary via bitcoins just now, and you spent those coins in the supermarket. The plaintext of the accounts of your incomes will inevitably be utilized by businesses to access the economic benefits, like charge you higher prices or target you with ads.

The possible solution to realize transaction privacy and delicate anonymity is to hide the amounts on each transaction. Cryptographically, it means to encrypt the plaintext on the transaction scripts. This feature, keeping the amounts transferred visible only to the participants, is also called Confidential Transaction, originally proposed by Adam Back on Bitcointalk in 2013 [27], and developed by Bitcoin Core Developer Gregory Maxwell [28,29] on Bitcoin Forum. The basic idea of [28] is to employ the Pedersen Commitment as a homomorphic tool to operate ciphers. Gregory then uses ring signature to sign on the Output amounts and the Input amounts, which implies a connection in its ring circle. The amounts in ciphertexts are proved to be correctly encrypted whenever they pass the ring verification. Instead of taking the commitments' sum to zero, Shen Noether proposed a modified method to blind sensitive information in the strongly decentralized anonymous cryptocurrency under Monero Community [30]. A new type of ring signature, called MLSAG, is used to hide the amounts, the origins and the destinations of the transactions. Another enlightening research related to our problem is Zerocoin [8,31]. Proposed by Matthew Green et al. Zerocoin system intends to confuse the sources of the transactions and change the mere pseudonymity into real anonymity, via creating a *decentralized laundry* by using zero-knowledge proof(ZKP). The additional benefit from its usage of zero knowledge proof is that the number of zerocoins is packaged into an envelope without leaking sensitive information. This concept is designed to be a new software layer on top of the Bitcoin protocol and it is finished under the coalition of several academic departments and the sponsor of Amazon et al. Above all, more detailed comparison will be shown in latter, and we focus on the problem of enhancing the privacy of the Bitcoin system.

Our contribution

To meet the needs of enhancing the transaction privacy in Bitcoin, we need a mechanism that can change the traditional way of transmission from the plaintext to the ciphertext, which improves the privacy and the security of the public network. The framework should be compatible with the existing Bitcoin system and can be applied simply and directly. We propose a framework by adding the homomorphic Paillier encryption system to cover the plaintext amounts in transactions, and the encrypted amounts will be checked by the Commitment Proof.

Our framework achieves the following requirements:

- To realize the concealment of the amounts in transactions by using the homomorphic Paillier encryption system. Encrypted transactions are like sealed envelopes of money which can be merged, split, or spent while simultaneously keeping the value transferred. Our scheme achieves delicate anonymity via hiding sensitive information on scripts.
- To realize the verification process of the amounts in ciphertext by using the Commitment Proof. On the one hand, the encrypted amounts in a transaction are proved to be in a specific interval via the commitment proof, which ensures that the amounts are always positive and avoids fraud negative number. On the other hand, we create a *dumb account* to receive the same amounts, which cannot be spent, to check the equality of the Input sum and the Output sum. The encrypted amounts can be added and subtracted correctly by employing the equality proof of two committed numbers.

- To be compatible with the original Bitcoin protocol. Without requiring new-constructed and advanced cryptographic assumptions, the modified scheme of the Bitcoin system can provide a strongly decentralized cryptocurrency without being utilized by attackers or privileged parties. Our scheme overcomes the problems like poor performance, mere pseudonymity, and low security.

Our scheme achieves the goals of delicate anonymity, verifiable correctness, and performable compatibility. Analysis shows that our framework is secure, practical and effective.

2. Preliminaries

2.1. Bitcoin system

Bitcoin is a decentralized electronic system which consists of three main technical components: Transactions, Consensus Protocol, and Communication Network [20]. These three main technical components naturally form the three progressive layers of the whole Bitcoin system: transactions, blocks, and blockchain.

Transactions.
Bitcoin works as a form of transaction, and each transaction contains an array of *Inputs* and an array of *Outputs*. The list of *Inputs* contains the previous output, the amounts, From address, and the ScriptSig. The *Outputs* contains the output index, the redeemed inputs, the amounts, To address and the ScriptPubKey. The ScriptSig and the ScriptPubKey, as users' private/public keys, specify the hash function of an ECDSA, an elliptic curve of secp256k1 [19]. The bitcoin addresses, the only identities in bitcoin, are computed via

$$Base58(SHA256(SHA256((RIPEMD160(SHA256(bPK))) \parallel version)))$$

by a public key, where Base58 represents an encoding way, RIPEMD160 and SHA256 represent different kinds of cryptographic hash functions. The amounts in transactions are plaintext and if valid, the sum of all transaction outputs is equal to the sum of inputs. All the details are collected in a transaction sheet, and then the entire transactions are hashed under SHA256 as its globally unique transaction ID. We can successfully trace a transaction through sequentially finding its transaction ID and correctly executing its ScriptSig and ScriptPubKey compared to the recorded hash.

Obviously, there are no inherent marks of the identities or the individual accounts linked with the bitcoin owners. Public key hashes functioned as the pseudonymously unique identities are referred to as *addresses*, and the ownership simply means knowing a private key which can be matched to its public key to redeem certain outputs. However, no real-world name or identity does not mean no threat. Attackers can link the bitcoin accounts with identities through the human activities in daily transactions, and that is the reason why we attach importance to the privacy in transactions.

Consensus protocol.
When transactions are completed, all the transactions must be published in a global and permanent log(public ledger), and every transaction can only be redeemed in the subsequent transactions. The log contains a series of transactions which are divided into a relatively small unit called *block* for its quick search, and each block is lined up in a proper sequence to form an irreversible chain called *blockchain*. We know that the transaction inputs refer to the previous transaction outputs by their transaction hash, and so as to blockchain. Each block head contains the hash of the previous block(other information such as address, timescale, nonce, Merkle root et al.) to keep its validity, and the block body contains several transactions as described above.

The validation of a block refers to the core innovation of the bitcoin system, a decentralized and pseudonymous protocol dubbed *Nakamoto consensus* [4]. The consensus prevents the system from *double spending* attack by its challenging computational puzzle mechanism *proof of work (PoW)*, which is implicitly connected with the *Byzantine Problem (BA)* to determine which party's block will be considered valid eventually. PoW solves the puzzle by finding a linked hash $H(B_{i-1}\|T_i\|R)$ (B_i, T_i short for block and transaction separately) that starts with d consecutive *zero* bits, and the standard strategy is to try random nonce R until found. The miners get rewards in proportional to their competing computational power and the difficulty of the puzzle is calibrated to keep in a relatively stable time interval — —10 min on average when generating a new block.

Communication network.
The bitcoin system bases on an ad-hoc peer-to-peer broadcast network to announce proposed block. Any newly joined node can connect to a random sample of neighbor nodes and generate a list of peers addresses. The well-connected peers will propagate information around each other by triggering a cascade relay messages [32]. If any latency occurs between the validation of blocks, temporary forks will accumulate by time. To avoid a frequent forks, a 10 min interval are selected by design. And to avoid a (group of) malicious miner to control the massive nodes, the broadcast network in decentralization is necessary.

2.2. Paillier cryptosystem

Our Bitcoin trading system exploits the Paillier encryption scheme to hide the plaintext of transactions amounts. The Paillier cryptosystem is a homomorphic scheme which is based on the composite residuosity class problem, and the scheme is provably secure under the appropriate intractability assumptions in the standard model. We review the definitions and the encryption/decryption process from [33].

Definition 1. For any $n \in \mathbb{Z}$, $\mathbb{Z}_n$ and $\mathbb{Z}_n^*$ are defined as:

$$\mathbb{Z}_n = \{x|x \in \mathbb{Z}, 0 \leq x < n\},$$
$$\mathbb{Z}_n^* = \{x|x \in \mathbb{Z}, 0 \leq x < n, \gcd(x, n) = 1\}.$$

Definition 2. For $\mathcal{S} = \{x < n^2 | x = 1 \bmod n\}$, $L(x)$ is defined as:

$$\forall x \in \mathcal{S}, \quad L(x) = \frac{x-1}{n}.$$

Definition 3. For RSA modulus $n = pq$ where p and q are large primes, $g \in Z_{n^2}^*$, and ε_g is defined as:

$$\mathbb{Z}_n \times \mathbb{Z}_n^* \longmapsto \mathbb{Z}_{n^2}^*$$
$$(x, y) \longmapsto g^x \cdot y^n \bmod n^2.$$

Definition 4 (*CRA*)**.** For ε_g described above and $w \in \mathbb{Z}_{n^2}^*$, we call the nth residuosity class of ω with respect to g the unique integer $x \in \mathbb{Z}_n$ for which there exists a certain $y \in \mathbb{Z}_n^*$ such that

$$\varepsilon_g(x, y) = \omega.$$

Here, the class of ω is denoted by $[\![\omega]\!]_g$.

The nth Residuosity Class Problem of base g, denoted by $Class[n, g]$, states that: for a given $\omega \in \mathbb{Z}_{n^2}^*$, compute $[\![\omega]\!]_g$ from ω.

Definition 5 (*DCRA*)**.** Decisional Composite Residuosity Assumption is defined as: Given $\omega \in \mathbb{Z}_{n^2}^*$, there is no polynomial time distinguisher for $y \in \mathbb{Z}_n^*$, where y satisfies

$$\omega = y^n \bmod n^2.$$

Definition 6 (*CCRA*)**.** Computational Composite Residuosity Assumption states that: given $c \in \mathbb{Z}^*_{n^2}$ and $n = pq$, there is no polynomial time algorithm to successfully compute m in

$$c = g^m r^n \bmod n^2$$

where $m \in \mathbb{Z}_n$ and $r \in \mathbb{Z}^*_{n^2}$.

The intractability hypothesis can be referred to as the *DCRA* throughout the cryptosystem, and the validity of the *DCRA* only depends on the choice of n. The basic construction is reduced to the hard problem *CCRA*, which uses the factorization as a trapdoor. We can get more detail in [33]. The encryption process states as follows:

KeyGen.

Set $n = pq$, compute $\lambda = \lambda(n) = lcm(p-1, q-1)$ and randomly select a base $g \in \mathbb{G}$ satisfying

$$\gcd(L(g^\lambda \bmod n^2), n) = 1$$

where p and q are large primes, $\mathbb{G}$ is a multiplicative group that $\mathbb{G} = \{w | w \in \mathbb{Z}^*_{n^2}\}$. The public key is (n, g) and private key is (p, q) or λ.

Encryption.

plaintext $\quad m < n$

select a random $\quad r < n$

ciphertext $\quad c = g^m \cdot r^n \bmod n^2.$

Decryption.

ciphertext $\quad c < n^2$

plaintext $\quad m = \dfrac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n.$

Paillier cryptosystem possesses additive homomorphic property as:

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m_1) \cdot \mathrm{Enc}_{pk}(m_2) \bmod n^2) = m_1 + m_2 \bmod n.$$

The inherent additive homomorphism makes the Paillier cryptosystem applied to various fields such as the design of voting protocols, the threshold cryptosystem, the secret sharing scheme watermarking, et al. The scheme also has the properties of being self-reducible and self-blinding, which deployed by our scheme. Except for these, the Paillier cryptosystem is convincingly secure under the chosen-plaintext attack in the standard model and it can efficiently perform in both encryption and decryption. It is perfectly suitable for hiding the transaction amounts on top of the Bitcoin system.

2.3. Proof of equality of two committed numbers

Committed number is widely used in electronic cash systems, group signatures, publicly verifiable secret sharing schemes, and zero-knowledge protocols. We sketch the definitions from [34].

Definition 7. Let $E = BC(x)$ be a commitment to a value $x \in [b_1, b_2]$. A proof of membership to an interval $[b_1, b_2]$ is a proof of knowledge to ensure the verifier that the prover knows x such that $E = BC(x)$ and that x belongs to $[b_x, b_y]$, an interval which contains $[b_1, b_2]$.

We set three security parameters: t, l and s, and set two large composite number n_1 and n_2 whose factorizations are unknown by the users. Let g_1 be an element of large order in $\mathbb{Z}^*_{n_1}$ and h_1 be an element of the group generated by g_1. And so as to h_2. Now Alice hold $x \in [0, b]$, we set

$$E = E_1(x, r_1) = g_1^x h_1^{r_1} \bmod n, \quad F = E_2(x, r_2) = g_2^x h_2^{r_2} \bmod n$$

are two *FO* commitment [35], where r_1 and r_2 are randomly selected over $\{-2^s n + 1, \dots, 2^s n - 1\}$.

The aim of Alice is to prove to Bob that she knows the value of x, r_1, r_2 and that the commitment values E and F hide the same secret x. The proof of equality of two committed numbers states as follows:

Protocol: $PK\{x, r_1, r_2 | E = E_1(x, r_1) \bmod n_1 \wedge F = E_2(x_2, r_2) \bmod n_2\}$.

1. Alice randomly selects $\omega \in \{1, \dots, 2^{l+t} b - 1\}$ and $\eta_1 \in \{1, \dots, 2^{l+t+s} n - 1\}$, $\eta_2 \in \{1, \dots, 2^{l+t+s} n - 1\}$, and computes
$$W_1 = g_1^\omega h_1^{\eta_1} \bmod n_1, \quad W_2 = g_2^\omega h_2^{\eta_2} \bmod n_2.$$
2. Alice computes $u = H(W_1 \parallel W_2)$.
3. Alice computes
$$D = \omega + ux, \quad D_1 = \eta_1 + ur_1, \quad D_2 = \eta_2 + ur_2$$
and sends Bob with (u, D, D_1, D_2).
4. Bob checks whether $u = u'$ where
$$u' = H(g_1^D h_1^{D_1} E^{-u} \bmod n_1 \parallel g_2^D h_2^{D_2} F^{-u} \bmod n_2).$$

From the above, we can easily verify the value u in step 2 and u' in step 4 are equal if the committed numbers E and F hide the same secret x. And during the communication, the secret is perfectly hidden in the committed number with irreversibility.

2.4. Committed number lied in a specific interval

The idea of checking whether a committed integer lies in a specific interval was first proposed in [36], and developed in [34,37]. *CFT Proof* [37] fulfills the aim to prove secret value lies in a specific range. However, it is obvious that the expansion rate compared to [34] is really large, which expands the interval from $[0, b]$ into $[-2^{120}b, 2^{120}b]$. The method in [34] can limit the committed number to a range $[a, b]$ with negligible deviation, but the protocol is too complicated to be applied in our scheme. As our requirement is to prove the encrypted amounts lies in $[0, m]$, we employs the method by Wu [38] which possesses both functionality and efficiency.

Let t, l and s be three security parameters, n be a large composite number whose factorization is unknown, g be an element of large order in $\mathbb{Z}^*_n$ and h be an element of the group generated by g. We set $E = E(x, r) = g^x h^r \bmod n$ is an *FO* commitment [35] to $x \in [a, b]$, where r is randomly selected over $\{-2^s n + 1, \dots, 2^s n - 1\}$. Here, $y = x - a$, $E_1 = g^{x-a} h^r = g^y h^r \bmod n$.

Protocol: $PK\{x, r | E = E(x, r) \bmod n \wedge x \in [a, b]\}$.

1. Alice sets $v = \alpha^2 y + \omega > 2^{t+l+s+T}$, where $\alpha \neq 0$, $\omega \in (0, 2^{s+T})$ are randomly selected. Set $r_3 - r\alpha^2 + r_1\alpha + r_2 \in [-2^s n + 1, \dots, 2^s n - 1]$, where $r_1, r_2, r_3 \in [-2^s n + 1, \dots, 2^s n - 1]$ are randomly selected, and compute
$$E_1 = g^{x-a} h^r = g^y h^r \bmod n,$$
$$E_2 = E_1^\alpha h^{r_1} \bmod n, \ E_3 = E_2^\alpha h^{r_2} \bmod n$$
$$F = g^\omega h^{r_3} \bmod n, \ V = g^v / E_3 = g^\omega h^{-r\alpha^2 - r_1\alpha - r_2} \bmod n.$$
Alice sends (V, E_2, E_3, F) to Bob.
2. Bob computes
$$E_1 = E(x, r)/g^a = g^y h^r \bmod n$$
$$U = g^v / E_3 = g^\omega h^{-r\alpha^2 - r_1\alpha - r_2} \bmod n.$$

3. Alice and Bob compute separately

$$PK1\{\alpha, r_1, r_2 : E_2 = E_1^{\alpha} h^{r_1} \bmod n \wedge E_3 = E_2^{\alpha} h^{r_2} \bmod n\} \quad (1)$$

$$PK2\{\omega, r^* : F = g^{\omega} h^{r^3} \bmod n \wedge U = g^{\omega} h^{r^*} \bmod n\} \quad (2)$$

$$PK3\{\omega, r_3 : F = g^{\omega} h^{r_3} \bmod n \wedge \omega \in [-2^{t+l+s+T}, 2^{t+l+s+T}]\} \quad (3)$$

where $r^* = -r\alpha^2 - r_1\alpha - r_2$.

4. Bob checks the correctness of PKi ($i = 1, 2, 3$) and that $v > 2^{t+l+s+T}$, which convinces Bob that $x > a$.
5. If we set $y = b - x$ at the beginning, repeat step 1-4 to prove $x < b$.

– completeness: If $x \in [a, b]$, the protocol fails with probability less than 2^{-l}.
– soundness: The forgery attack succeeds with probability $3 \times 2^{-t+1}$.
– zero-knowledge: Statistically zero-knowledge in the random oracle.
– proof content: $x \in [a, b]$.
– expansion rate: $\delta = 1$.

3. System model

3.1. Problem statement

Transaction privacy covers numerous aspects in Bitcoin and we focus on its plain amounts on scripts. It is widely acknowledged that the Bitcoin system is anonymous by its unrelated association between the addresses and the identities. However, a potential threat is the exposure of amounts in transactions. The plain amounts may leak sensitive information to business analysis or individual privacy. The advertisers could deduce some valuable information by analyzing the absolute values of amounts under special situations and then cast ads on them. We intend to hide the plain amounts with a homomorphic cryptosystem and ensure that it can be smoothly operated in the Bitcoin system. We on the one hand, propose a secure method that achieves confidential transaction to prevent attacks from attackers; On the other hand, the scheme is perfectly compatible with the originally existing Bitcoin system by providing a verification mechanism to check the ciphered amounts.

3.2. The system model

Our scheme employs the Paillier cryptosystem to hide the amounts in the transactions. To successfully achieve transaction correctness, we create a *dumb account* to receive the same amounts of bitcoins for the verification of the positive-number check and the Input-sum&Output-sum check. We briefly sketch our scheme with highlighted steps in Fig. 1 and describe it as follows:

$(pk_i, pk_d) \leftarrow KeyGen(t, l, s, T)$.
The inputs are security parameters for encryption and verification separately. The algorithm outputs the public key $pk_i(n_i, g_i)$ of every receiver i with the private key sk_i and the public key $pk_d(n_d, g_d)$ of the system for verification without any private key.

$(m) \leftarrow InSum(c_{in}, reward)$.
The previous transaction output becomes the present transaction input. Here we check the situation whether the transaction is from a newly confirmed block. If yes, we add an extra encouragement *reward* from the Bitcoin system to the original input encrypted amounts c_{in}, and the algorithm outputs m representing the whole bitcoins that Alice possesses. Note that m is also the secret hidden in the committed numbers to be verified in the latter steps.

$(c_i, c_{id}) \leftarrow Encrypt(m_i, pk_i, pk_d)$.
Whenever a trade between the sender and the receivers occurs, the amounts m_i in plaintext are simultaneously encrypted by pk_i and pk_d. The algorithm outputs c_i to the receivers in the transaction layer and c_{id} to the *dumb account* in the verification layer.

$(0/1) \leftarrow Verify(V_\alpha, V_\beta, \prod c_{id}, m, m_i)$.
The verification layer intends to check that the hidden amounts are positive and that the input-sum and output-sum are equal. The verification firstly confirms the positiveness of the secrets in the commitments in *Verify-I*. Alice makes commitments and convinces receivers that the encrypted amounts are always positive by the Commitment Proof (refer to Section 2.4). Secondly, we focus on the equality proof of the Input-sum and the Output-sum (refer to Section 2.3) in *Verify-II*. The algorithm takes inputs as the parameters $V\alpha(W_\alpha, D_\alpha)$, $V_\beta(W_\beta, D_\beta)$ to generate two committed numbers containing the same secret $m = \sum m_i$. Alice sends the committed numbers E and F to the *dumb account* to check the validity. Then, for the *dumb account*, the system operates the received ciphertexts c_{id} by multiplying as $\prod c_{id}$, and verifies the equality of F and H. The verification employs $E = F = H$ to confirm its sum equality. If the algorithm returns 1, the protocol runs into the next step.

$(m_i) \leftarrow Decrypt(0/1, sk_i, c_i)$.
As the verification goes well (input 1), the encrypted amounts c_i is sent to the receivers in the transaction layer. The receivers get the ciphertext c_i and decrypt it with sk_i. The received amounts automatically operate in the system to form the subsequent input under the same mechanism.

$(T_{Alice}) \leftarrow Broad(c_{in}, c_i, addresses)$.
When the verification is correctly verified and the transmission of the trades is successfully finished, a transaction is completely accomplished. The transactions will be then broadcast to the Internet to wait for being confirmed. The script of the transaction T_{Alice} only presents the information of c_i and *address* which are delicately anonymous.

3.3. Security goals

Our scheme is designed to achieve the following security goals and the requirements are based on top of the existing Bitcoin system:

Robust transaction.
Firstly, our scheme aims to be compatible with the existing Bitcoin system and achieve the smooth transaction. The blockchain-based cryptocurrencies all confront the common problems like double-spending problem and inconsistency problem, which are fatal to the persistence and steadiness of the system. It is acknowledged that the mature Bitcoin system has witnessed its success in overwhelming problems in this field. So our scheme plans to embed the Paillier cryptosystem into the Bitcoin system. As for the induced verification process in ciphertext, we employ two kinds of zero-knowledge proofs to address the verification difficulties in positiveness and equality.

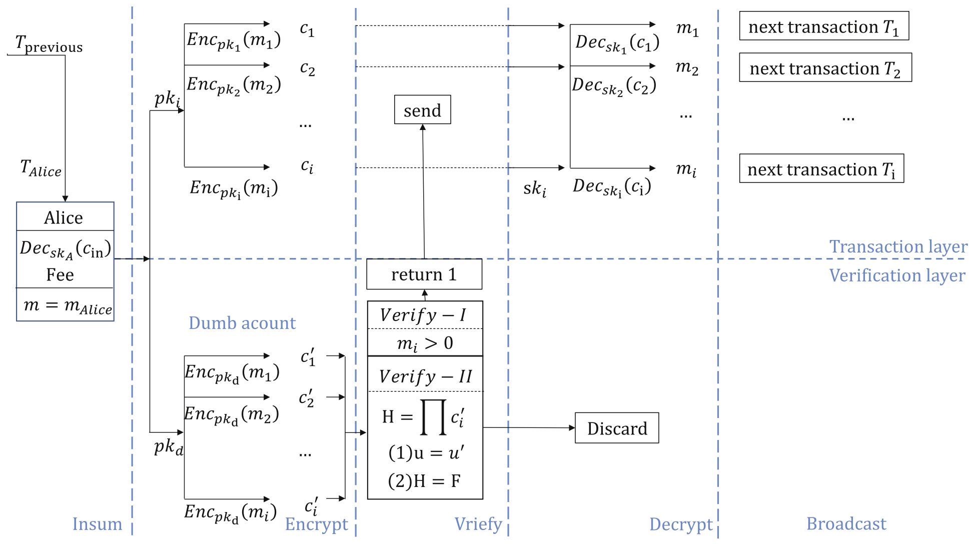

Fig. 1. Scheme model.

Transaction privacy.

Secondly, our scheme aims to achieve delicately confidential transaction and assure every transaction correctly executes. The transaction amounts in plaintext might leak unpredictably massive information during the daily trading. The plain and recognizable details on scripts weaving with the historic record on blockchain make individuals exposed under the whole Internet world. It is important to improve the privacy and protect our secrete in daily transactions. Our scheme plans to employs the public-key system to encrypt the amounts and hide the sensitive information in transactions. The homomorphic property of the Paillier cryptosystem enables our system operate in ciphertext. So we aims to achieve transaction by using cryptographic methods.

Attacks resistance.

Finally, our scheme aims to prevent several types of attacks. Since the transaction is publicly exposed to the Internet, the attackers may conduct any kind of attacks to forge transactions or destroy the systems. The attacks involves both active attacks and passive attacks, and different attacks need different methods to overcome. After a brief comparable analysis, We plans to improve the security level both in the unforgeability of Bitcoin and of transactions. Due to its inherent mechanism of the confirmation in the Bitcoin system, there is no record with the creation of bitcoin out of thin air, so that it is nearly impossible for the first forgery. And because of the avalanche-effect of the hash function and the hard mathematic problems in the Paillier cryptosystem, there is negligible possibility for the second forgery, neither.

3.4. Adversary model

The framework may suffer from both active attacks and passive attacks shown as follows:

Active attack.

Since each transaction in the Bitcoin system is broadcasted to the P2P network, the attackers may deliberately destroy the system or maliciously forge transactions in all possible ways.

- **Tampering attack.** The attackers may tamper the bitcoin transactions of the bitcoin addresses, amounts and other information after signing. They broadcast transactions to the P2P network waiting for validation. To prevent this attack, the transaction generated by our scheme should be rejected as soon as found.
- **Overlay attack.**The attackers may maliciously forge a ciphered amount to replace its original sending cipher since it can be homomorphically operated in the Paillier cryptosystem under the same receiver's pk_i which is public. Traders should take more attention to this kind of attack since the operated ciphertext is unrecognizable at the first glance. Our scheme should prevent such attack.
- **Double-spending attack.** The attackers may spend bitcoins twice before the transactions are confirmed. Since the confirmation needs the most computing power accumulated by the distributed nodes, if a group of attackers hold more than half percent computing power, the system of Bitcoin might be controlled by them. But fortunately, there are no groups processing such computing power that may destroy the system. Our scheme should be compatible with the Bitcoin system so that we can prevent this attack.

Passive attack.

Our scheme may be confronted with passive attacks including eavesdrop attack and traffic analysis. The attackers intend to extract secret information from the traders by monitoring their communications, or analyzing the traffic data of their transactions through the internet. The system may occasionally be exposed to some sensitive information in public and our scheme is trying to hide all of them which are at risk.

4. Transaction privacy with Paillier

4.1. The concrete scheme

In this section, we investigate the basic scenario on confidential transaction in the Bitcoin system. We propose our concrete scheme which solves four technical problems:

- To hide the amounts in a transaction, technically, we use the Paillier cryptosystem to encrypt the plain amounts into cipher. The ciphertexts replace the original plain amounts and only the receivers who own the private keys are able to decrypt them.
- To solve the problem of multiparty-trading under one public cryptosystem, technically, we create the *dumb account* (a kind of interlayer) as a bridge between the sender and different receivers in each transaction. With the same public system parameters but none of private key, the special

account on the one hand, circumvents the obstacle that the Paillier ciphertexts cannot be operated via its additive homomorphism under different *pk*s from different traders; On the other hand, the design ensures that the *dumb account* does not allow to decrypt and spend money.

- To verify the positiveness of the encrypted amounts in the transaction, technically, we employ a proof of committed numbers lied in a specific interval. We have compared several proof methods and select one possessing both function and efficiency. The verification plays an important position in transactions, since it can prevent the deception inside from the system.
- To verify the equality of the Input-sum and the Output-sum in the transaction, technically, we employ the proof of equality of two committed numbers which can prove two *blind* numbers contain the same secret value. To make it perfectly suitable for cryptosystem, which means the operated ciphertext becomes one of committed number, we link it due to its similarity to the basic construction, by choosing *dumb account*'s generator g_d in the encryption of the Paillier cryptosystem equal to the generator g_β in the verification, and select random r_d equal to h_β in the encryption.

The scheme states as follows:

$(pk_i, pk_d) \leftarrow KeyGen(t, l, s, T)$.

For each different receiver (distributed nodes) i, the system generates two large primes p_i and q_i. The secret key is $sk_i = \lambda_i$ where $\lambda_i = \text{lcm}(p_i - 1)(q_i - 1)$. And the public key is $pk_i = (n_i, g_i)$, where $n_i = p_i q_i$ and $g_i \in \mathbb{Z}_{n_i^2}$ such that $g_i \equiv 1 \bmod n_i$.

Simultaneously, the system generates another public key $pk_d = (n_d, g_d)$ for each transaction as the parameter of the *dumb account*, where n_d is a large safe composite number hard to be factorized. Remarkably, there is no private key for the *dumb account* to decrypt or spend money.

As for the verification, the system generates $V_\alpha(g_\alpha, h_\alpha)$ and $V_\beta(g_\beta, h_\beta)$, where g_α is an element of large order in $\mathbb{Z}^*_{n_\alpha}$ and h_α is an element of the group generated by g_α such that both the discrete logarithm of g_α in base h_α and the discrete logarithm of h_β in base g_α are unknown to the sender. The same does to g_β and h_β.

Notably, since the *dumb account* and the verification are executed in the same layer, and they are both generated by the system in each transaction, we can set $g_\beta = g_d$ and $n_\beta = n_d^2$ so as to make the operated ciphertext under the Paillier cryptosystem become a committed number which can be verified.

$(m) \leftarrow InSum(c_{in}, reward)$.

We assume the sender called Alice and there are two cases for the Input-sum:

- If Alice is a normal receiver, her input-sum equals to what she receives in precedent transactions:

  $m = m_{in} = \text{Dec}_{pk_{Alice}}(c_{in})$.

- If Alice is a successful miner (or a member of a miner group) of a new block, she will get extra rewards for 50 bitcoins in plaintext from the system. And referring to additive homomorphic property (9), the input-sum can be calculated as

  $m = m_{in} + 50 = \text{Dec}_{pk_{Alice}}(c_{in} \otimes 50)$

  where c_{in} means the inputs of whoever sends to Alice.

$(c_i, c_{id}) \leftarrow Encrypt(m_i, pk_i, pk_d)$.

We employ the Paillier cryptosystem to hide the plain amounts $m_1, m_2, \ldots, m_i$ into the ciphertexts $c_1, c_2, \ldots, c_i$ under the different $pk_1, pk_2, \ldots, pk_i$ from the receiver i. We encrypt each amounts under the system's pk_d in parallel so as to verify the equality of the inputs and the outputs. Rather than being sent into the receivers' accounts, the encrypted bitcoins under the system pk_d are sent into the *dumb account* that cannot spend money (for no sk_d). This *double encrypted layer* mechanism, containing the transaction layer and the verification layer, is pivotal for our scheme.

In the transaction layer, the system encrypts amounts under different pk_i from each receiver i:

$c_i = \text{Enc}_{pk_i}(m_i) = g_i^{m_i} r_i^{n_i} \bmod n_i^2$.

In the verification layer, the system encrypts amounts under the same pk_d from each transaction:

$c_{id} = \text{Enc}_{pk_d}(m_i) = g_d^{m_i} r_d^{n_d} \bmod n_d^2$.

Here, we select the random number $r_d = h_\beta$, which is an element of the group generated by $g_\beta \in \mathbb{Z}^*_{n_\beta}$

The common point between two layers is that they have encrypted the same amount m_i, and obviously, the difference is that: the first layer uses different pk_is but the second layer uses the same pk_d, so as to achieve the homomorphic operation of the Paillier cryptosystem. The inherent same m_i ensures we send correct and exact amounts to the receiver i.

$(0/1) \leftarrow Verify(V_\alpha, V_\beta, \prod c_{id}, m, m_i)$.

In the encryption process, Alice sends her bitcoins to the receivers in the transaction layer under pk_i, and sends the same amounts to the *dumb account* in the verification layer under pk_d. Now in the verification layer, we will check the correctness of the transaction amounts in two steps:

I: The amounts m_i inside the ciphertexts c_i are always positive or satisfy $m_i \in [0, m]$ (refer to Section 2.4).

II: The output-sum $\sum m_i$ inside cooperated cipher $\prod c_i$ equals to input-sum m in the commitment (refer to Section 2.3).

Verify-I.

Each transaction amount m_i by Alice is encrypted into the ciphertext c_i, we employ the commitment proof to prove the secret value m_i lied in $[0, m]$. Remarkably, Alice does not want the receivers to know her amounts m, so she can randomly choose a number b which is less than m. The proof of $m_i > 0$ is necessary. However, it is redundant to prove $m_i < m$ since we need to check the output-sum is equal to the input-sum. If a single m_i is more than the Input-sum m while each $m_i > 0$, it is impossible to make In/Out equal. Therefore, we give the verification of $m_i > 0$. Noted that, the g, h, n in the above steps can be the same as g_d, h_d, n_d, for simplicity, we just write as g, h, n. Alice makes commitments $E_{i0}(m_i, r) = g^{m_i} h_r \bmod n$ for each m_i, also for simplicity, we just write $E_{i0}(m_i, r), E_{i1}, E_{i_2}, E_{i3}, F_i, V_i$ in the steps as $E_0(m_i, r), E_1, E_2, E_3, F, V$ to represent a typical one of them.

To achieve our goals, Alice and Bob interact as follows:

1. Alice sets $v = \alpha^2 y + \omega > 2^{t+l+s+T}$, where $\alpha \neq 0$, $0 < \omega \leq 2^{s+T}$ are randomly selected. Then Alice sets $r_3 - r\alpha^2 + r_1\alpha + r_2 \in [-2^s n + 1, \ldots, 2^s n - 1]$, where $r_1, r_2, r_3 \in [-2^s n + 1, \ldots, 2^s n - 1]$ are randomly selected, and computes

   $E_1 = g^{m_i - a} h^r = g^y h^r \bmod n$

   $E_2 = E_1^\alpha h^{r_1} \bmod n,\ \ E_3 = E_2^\alpha h^{r_2} \bmod n$

   $F = g^\omega h^{r_3} \bmod n$

   $V = g^v / E_2 = g^\omega h^{-r\alpha^2 - r_1\alpha - r_2} \bmod n$.

   Alice sends (V, E_2, E_3, F) to Bob.

2. Bob computes

$$E_1 = E_0(m_i, r)/g^a = g^y h^r \bmod n$$

$$V = g^v/E_3 = g^\omega h^{-r\alpha^2 - r_1\alpha - r_2} \bmod n.$$

3. Alice and Bob compute separately

$$PK1 \quad \{\alpha, r_1, r_2 : E_2 = E_1^\alpha h^{r_1} \bmod n \wedge E_3 = E_2^\alpha h^{r_2} \bmod n\} \tag{4}$$

$$PK2 \quad \{\omega, r^* : F = g^\omega h^{r^3} \bmod n \wedge V = g^\omega h^{r^*} \bmod n\} \tag{5}$$

$$PK3 \quad \{\omega, r_3 : F = g^\omega h^{r_3} \bmod n \wedge \omega \in [-2^{t+l+s+T}, 2^{t+l+s+T}]\} \tag{6}$$

where $r^* = -r\alpha^2 - r_1\alpha - r_2$.

4. Bob checks the correctness of PK$i(i = 1, 2, 3)$ and that $v > 2^{t+l+s+T}$, which convinces Bob that $x > a$ (we set $a = 0$).
5. For each m_i, repeat step 1-4 to prove $m_i > 0$ $(i \in \{1, 2, \dots, i\})$.

The steps will be iterated for i times for every m_i. If there were any one of them failed, the system returns 0 and stops. If all succeed, the system returns 1, and then goes to the next verification process.

Verify-II.

Alice knows her input-sum m and her distribution to each receiver m_i which satisfies $m = m_1 + m_2 + \cdots + m_i = \sum m_i$. Now Alice makes two committed number

$$E = E_\alpha(m, r_\alpha) = g_\alpha^m h_\alpha^{r_\alpha}, \; F = E_\beta\left(\sum m_i, r_\beta\right) = g_\beta^m h_\beta^{r_\beta},$$

where $r_\alpha \in \{-2^s n+1, \dots, 2^s n-1\}, r_\beta = n_d \in \{-2^s n+1, \dots, 2^s n-1\}$, and s is a security parameter. If we want to verify the received ciphertexts whether equal to the Input-sum from Alice in the *dumb account*, he checks that:

(1) The secret $m = \sum m_i$ in E and F from Alice are identical.
(2) The operated ciphertext $H = \prod c_{id}$ is equal to the committed number F.

To achieve the first step:

1. Alice picks random $\omega \in \{1, \dots, 2^{i+t}b - 1\}$, $\eta_\alpha \in \{1, \dots, 2^{l+t+s}n - 1\}$, $\eta_\beta \in \{1, \dots, 2^{l+t+s}n - 1\}$, and computes

$$W_\alpha = g_\alpha^\omega h_\alpha^{\eta_\alpha} \bmod n_\alpha, \quad W_\beta = g_\beta^\omega h_\beta^{\eta_\beta} \bmod n_\beta.$$

2. Alice computes $u = H(W_\alpha \parallel W_\beta)$.
3. Alice computes

$$D = \omega + um, \quad D_\alpha = \eta_\alpha + ur_\alpha, \quad D_\beta = \eta_\beta + ur_\beta$$

and sends $(u, D, D_\alpha, D_\beta)$ to the *dumb account*.

4. In the *dumb account*, we check whether $u = u'$, where

$$u' = H(g_\alpha^D h_\alpha^{D_\alpha} E^{-u} \bmod n_\alpha \parallel g_\beta^D h_\beta^{D_\beta} F^{-u} \bmod n_\beta).$$

If the first step succeeds, then we check the second step:

1. For the *dumb account* we compute the ciphertexts:

$$H = \prod c_{id} = c_1' c_2' \dots c_i' = g_d^{m_1' + m_2' + \cdots + m_i'} r_d^{n_d} = g_d^{\sum m_i'} r_d^{n_d} \bmod n_d^2.$$

2. From the above, we can randomly select $r_d = h_\beta$ in encryption and $r_\beta = n_d$ in verification. And from KeyGen we set $n_\beta = n_d^2$, and $g_d = g_\beta$. So we can see:

$$H = g_d^{\sum m_i'} r_d^{n_d} \bmod n_d^2$$
$$F = g_\beta^{\sum m_i} h_\beta^{r_\beta} \bmod n_\beta = g_d^{\sum m_i} r_d^{n_d} \bmod n_d^2.$$

3. Check whether H equals to F. If no, we reject the transaction; and if yes, we returns 1 for the next process.

$(m_i) \leftarrow$*Decrypt*$(0/1, sk_i, c_i)$.

As described in the last process, if the verification layer returns 1, it sends the encrypted ciphers c_i to the receivers with inherently correct amounts inside. When the receivers receive the ciphertexts, they can use their sk_i to decrypt them:

$$m_i = \frac{L(c_i^{\lambda_i} \bmod n_i^2)}{L(g_i^{\lambda_i} \bmod n_i^2)} \bmod n_i$$

where $L(x) = \frac{x-1}{n}$ and $x \in \mathcal{S}_n = \{u < n^2 | x = 1 \bmod n\}$.

When the receivers decrypt ciphertexts and assure the amounts are exactly what they want, one transaction has been finished. The amounts transferred to receivers can be seen as the next inputs, so the same process can be repeated in every transaction. It is noteworthy that whenever the transaction is finished, the *dumb account* will be discarded automatically, since it exists only in the verification process and it cannot spend any bitcoins received.

$(T_{Alice}) \leftarrow$*Broad*$(c_{in}, c_i, addresses)$.

After correctly verified by each receiver, the transaction can be broadcast to the P2P network to form a new block. The scripts hide the original plain amounts into ciphertext and the contents on the scripts are unrecognizable strings c_{in} from the last transaction and c_i toward targeted *address*. We remark this script as T_{Alice}, and the same pattern is suitable for all others.

4.2. Correct analysis

Transaction correctness.

Firstly, we give the brief correctness analysis of the Paillier cryptosystem used both in the transaction layer and the verification layer, referring to [33]:

Lemma 1. *For any* $\omega \in \mathbb{Z}_{n^2}^*$, $L(\omega^\lambda \bmod n^2) = \lambda[\![\omega]\!]_{1+n} \bmod n$.

Since $1 + n \in \mathbb{G}$, so $[\![g]\!]_{1+n} = [\![1 + n]\!]_g^{-1} \bmod n$ is revertible which results in that $L(g^\lambda \bmod n^2)$ is revertible modulo n. Therefore, for any $g \in \mathbb{G}$ and $\omega \in \mathbb{Z}_{n^2}^*$, we compute

$$\frac{L(\omega^\lambda \bmod n^2)}{L(\omega^\lambda \bmod n^2)} = \frac{\lambda[\![\omega]\!]_{1+n}}{\lambda[\![g]\!]_{1+n}} = \frac{[\![\omega]\!]_{1+n}}{[\![g]\!]_{1+n}} = \frac{[\![\omega]\!]_g [\![g]\!]_{1+n}}{[\![g]\!]_{1+n}} = [\![\omega]\!]_g \bmod n.$$

From the above, we can correctly decrypt the c_i to acquire the original accounts m_i for the traders.

Verification correctness.

Secondly, we give the correctness analysis of the verification layer, and we state the *Verify-I* and the *Verify-II* in sequence as follows:

Verify-I.

Bob convinces from PK1 (4) that

$$E_1 = g^y h^r \bmod n$$
$$E_2 = E_1^\alpha h^{r_1} = g^{\alpha y} h^{\alpha r + r_1} \bmod n$$
$$E_3 = E_2^\alpha h^{r_2} = g^{\alpha^2 y} h^{\alpha^2 r + \alpha r_1 + r_2} \bmod n.$$

Bob convinces from PK2 (5) that

$$F = g^\omega h^{r_3} \bmod n$$
$$V = g^v / E_3 = g^\omega h^{-r\alpha^2 - r_1\alpha - r_2} \bmod n$$

so that

$$\begin{aligned} g^v &= VE_3 \\ &= g^\omega h^{-r\alpha^2 - r_1\alpha - r_2} g^{\alpha^2 y} h^{\alpha^2 r + \alpha r_1 + r_2} \\ &= g^{\alpha^2 y + \omega} h^0 \\ &= g^{\alpha^2 y + \omega} \bmod n \end{aligned}$$

and we get

$$v = \alpha^2 y + \omega \bmod \varphi(n)$$

where $\varphi(n)$ is Totient function.

Here, Alice knows neither the factorization of n nor the function $\varphi(n)$. From the strong-RSA assumption, there only exists $v = \alpha^2 y + \omega$.

Bob has verified that $v > 2^{t+l+s+T}$ and convinces from PK3 (6) that

$$\omega \in [-2^{t+l+s+T}, 2^{t+l+s+T}],$$

so we get

$$y > 0.$$

Otherwise, if $y < 0$, we obtain

$$v = \alpha^2 y + \omega \le \omega \le 2^{t+l+s+T}$$

which **contradicts** to $v > 2^{t+l+s+T}$ in step 1&4 of the *Verify-I* .

From the above, we can prove $y = m_i - a > 0$, and $m_i > a$, where $a = 0$. The correctness analysis can be applied to each m_i for $i \in \{1, 2, \ldots, i\}$. And if it turns true, we go into the next:

Verify-II.

From the verification steps [34], we can see

$$u = H(W_\alpha \| W_\beta) = (g_\alpha^\omega h_\alpha^{\eta_\alpha} \bmod n_\alpha \| g_\beta^\omega h_\beta^{\eta_\beta} \bmod n_\beta).$$

And if two committed numbers E and F are equal, then for the *dumb account*, we have:

$$\begin{aligned} u' &= H(g_\alpha^D h_\alpha^{D_\alpha} E^{-u} \bmod n_\alpha \parallel g_\beta^D h_\beta^{D_\beta} F^{-u} \bmod n_\beta) \\ &= H(g_\alpha^{um+\omega} h_\alpha^{ur_\alpha+\eta_\alpha} (g_\alpha^m h_\alpha^{r_\alpha})^{-u} \bmod n_\alpha \parallel \\ &\quad g_\beta^{um+\omega} h_\beta^{ur_\beta+\eta_\beta} (g_\beta^m h_\beta^{r_\beta})^{-u} \bmod n_\beta) \\ &= H(g_\alpha^{um+\omega-um} h_\alpha^{ur_\alpha+\eta_\alpha-ur_\alpha} \bmod n_\alpha \parallel \\ &\quad g_\beta^{um+\omega-um} h_\beta^{ur_\beta+\eta_\beta-ur_\beta} \bmod n_\beta) \\ &= (g_\alpha^\omega h_\alpha^{\eta_\alpha} \bmod n_\alpha \parallel g_\beta^\omega h_\beta^{\eta_\beta} \bmod n_\beta) \\ &= u. \end{aligned}$$

And then, we focus on the operated ciphertext H from the *dumb account* and the committed number F from Alice:

$$\begin{aligned} H &= \prod c_i' \\ &= g_d^{\sum m_i'} r_d^{n_d} \bmod n_d^2 \\ F &= g_\beta^{\sum m_i} h_\beta^{r_\beta} \bmod n_\beta \\ &= g_d^{\sum m_i} r_d^{n_d} \bmod n_d^2 \end{aligned}$$

where $r_d = h_\beta$, $r_\beta = n_d$ are randomly selected , and $n_\beta = n_d^2$, and $g_d = g_\beta$ are generated above. If H equals to F where

$$H = g_d^{\sum m_i'} r_d^{n_d} \bmod n_d^2 = g_d^{\sum m_i} r_d^{n_d} \bmod n_d^2 = F$$

we can clearly conclude that:

$$\sum m_i' = \sum m_i = m$$

which means that the Input-sum equals to the Output-sum, and the verification layer turns to be correct.

4.3. Security analysis

Active attack.

Our scheme can resist the active attacks. The active attack means an attack can actively destroy the transaction in different types, here we state our security analysis:

- **Tampering attack.** Our scheme uses the public-key cryptosystem to encrypt information, and only the receiver with private key can acquire the legally usable bitcoins. If the tampering by attackers occurs to the ciphertexts (of the transaction amounts), failed decryption will make transactions be discarded. And if the tampering occurs to information like *address*es and *pk*s, the avalanche effect of hash algorithm will rapidly reject the transaction. So the scheme can resist the information tampering attack.
- **Overlay attack.** Overlay attack means that the attacker adds an forgery encrypted amount c_f to the original encrypted amount c_i under the receiver's pk_i. On the one hand, if the attackers add another amount to the transaction, the Input-sum and the Output-sum become unequal, so that the verification will be rejected from the trade. On the other hand, our scheme bases on bitcoin system and every transaction is embedded with an Time-Stamp to mark the uniqueness. Different inputs under the trader i can be distinguished to the different transactions. So the scheme can resist overlay attack.
- **Double-spending attack.** Double-spending means the attackers spends a sum of bitcoins twice to acquire extra amounts. The basic construction of bitcoin system employs the Time-Stamp and the Proof-of-Work mechanism to effectively solve this problem, so the transaction cannot be authenticated. What we want stresses is our creation *dumb account* draws lessons from this kind of attack. We spend our bitcoins twice in each transaction with the same amounts inside ciphertexts. The critical point is our *dumb account* has no private key to spend money latter, so it is going to be discarded as soon as the transaction finished. As for the real transactions, we have not created any extra bitcoins at all.

Passive attack.

Our scheme can resist passive attacks. The passive attacks usually contain information monitoring and traffic analysis. In an original transaction, the attacker can access two kinds of information from the traders: the one is *address*es, the other is *amount*s. The address is a hash signature from the trader which cannot be reversible. So the only potential information monitored by attackers is the plaintext amounts in transactions, and that is what our scheme focused on. The Paillier cryptosystem has been used to encrypt and protect the amounts shown on the scripts, and what we can see is an unrecognized string which can only be readable to the receiver with private key. The security of ciphers rely on chosen encryption system, and as referred to [33], we can conclude that our scheme, with the Paillier cryptosystem which bases on modular arithmetics, is provably secure under chose-plaintext attack in the standard model. As for the traffic analysis, all the data was recorded on the transactions and then was broadcast to the P2P network. Remarkably, in our verification layer, the ciphertexts encrypted under pk_d is not shown on the script and in further, there is no private key for anyone to decrypt it. So when the verification has been confirmed, this part of money is going to be discarded. Passive attacks cannot effectively destroy the transaction as described above.

4.4. Property analysis

Our scheme processes properties as follows:

Additive homomorphism.

The additive homomorphism of the Paillier cryptosystem enables the following:

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m_1)\cdot \mathrm{Enc}_{pk}(m_2) \bmod n^2) = m_1 + m_2 \bmod n \tag{7}$$

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m)^k \bmod n^2) = km \bmod n \tag{8}$$

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m_1)\cdot g^{m_2} \bmod n^2) = m_1 + m_2 \bmod n. \tag{9}$$

There are three additive homomorphic expressions of the Paillier system which corresponds to three trading situations:

1. Input to Outputs with different encrypted amounts we use (7).
2. Input to Outputs with same encrypted amounts we use (8).
3. Input to Outputs with an encrypted amount and a plain amount we use (9).
   When a new block is validated, the miner obtains rewards 50 bitcoins (or shared for less) in plain, and then the input(from previous output) executes a ciphertext with plaintext.

Self-blinding.

Any ciphertext can be publicly changed into another one without affecting the plaintext, which states as: $\forall m \in \mathbb{Z}_n$ and $r \in \mathbb{N}$

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m) r^n \bmod n^2) = m \bmod n$$
$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m) g^{nr} \bmod n^2) = m \bmod n.$$

Although there are few situations needed to replace the cipher which is correctly encrypted, some special cases might call for that requirement to improve the security level.

Zero-knowledge.

Our scheme can reach an agreement on verifying the equality of the Input-sum and the Output-sum without leaking any secret. The information flown through the system is the commitments E, F and u, W_α, W_β which are unreadable for attackers. The amounts are completely hidden in the committed numbers like their hiding in ciphertexts. Our verification proves that two committed numbers hide the same secret, and it makes a linkage to the operated ciphertexts to establish a way for proof under the unrelated parties (the Input-sum and the Output-sum).

4.5. Efficiency analysis

We analyze the complexity of our scheme. We set $\tau_m, \tau_M, \tau_E, \tau_H$ to represent the unit of multiplication operating time, modular multiplication time, modular exponentiation time, and hash function operating time, separately. The detailed evaluations are shown in Table 1.

Note that every unit of τ is usually small in practice, so that our scheme is theoretically efficient. However, these estimates are purely indicative, and are not resulted from an actual implementation.

Table 1
Performance evaluation.

Complexity	Unit of time
Key generation	$(i+2)\tau_m$
Encryption	$2i\tau_M + 4i\tau_E$
Verification	$(2i+3)\tau_m + (7i+8)\tau_M + (12i+14)\tau_E + 2\tau_H$
Decryption	$2\tau_m + 2i\tau_E$

To save the computational time in further, there are two methods to speed up the Paillier cryptosystem referred to [33]:

- Employ the *Pre-processing techniques* for exponentiating a constant base.
  The second computation r^n or $g^{nr} \bmod n^2$ in encryption and the constant parameter $L(g^\lambda \bmod n^2)^{-1} \bmod n$ in decryption can be computed in advance.
- Use the *Chinese Remainder Theorem(CRT)* for decryption.
  The CRT can efficiently reduce the decryption workload by employing the L_p and L_q where

  $$L_p(x) = \frac{z-1}{p}, \quad L_q(x) = \frac{x-1}{q}.$$

  Therefore, the decryption can be accelerated by separately computing divisible massage and recombing modular residues :

  $$m_p = L_p(c^{p-1} \bmod p^2) l_p \bmod p$$
  $$m_q = L_q(c^{q-1} \bmod q^2) l_q \bmod q$$
  $$m = \mathrm{CRT}(m_p, m_q) \bmod pq$$

  with the pre-computations

  $$l_p = L_p(g^{p-1} \bmod p^2)^{-1} \bmod p$$
  $$l_q = L_q(g^{q-1} \bmod q^2)^{-1} \bmod q.$$

4.6. Comparison analysis

Here, we briefly analyze the Confidential Transaction scheme and the Zerocoin system as follows:

Confidential Transaction (CT) was originally proposed by Adam Back [27], and developed by Gregory Maxwell [28,29]. The basic idea of the CT is to employ the Pedersen Commitment as a homomorphic tool to operate the plain amounts. It uses the ring signature as the zero knowledge proof to check the validity of the signed Output amounts and Input amounts. The inherent conception of the ring signature represents a connection between the senders and the receivers, which can act as an envelope to seal the plain amounts. In further, instead of taking the commitments' sum to zero, Shen Noether proposed a modified method to blind the sensitive information and describes a strongly decentralized anonymous payment scheme [30]. Shen employs a modified type of the ring signature called *Multi-layered Linkable Spontaneous Anonymous Group signature (MLSAG)*, to hide the sensitive information containing amounts, origins and destinations in the transactions. The development of the CT enlightens us with its homomorphic method and the commitment verification thought.

However, the CT scheme faces several threats which cannot be ignored. For one, the scheme requires a given pair of public key and amounts (P, A) to be used in the ring signature with other public keys while they have the same amounts. There may be a potential pairs (P', A') available with $A = A'$ in ring signature, which leads that the potentially anonymous set is perhaps different as expected. For other, the scheme confronts the blockchain analysis which bases on the amounts sent in a specific given transaction. Attackers can narrow down the possibilities of the sender if they know some specific amounts from the begin.

Zerocoin [8] is based on the Bitcoin system, aiming to provide the strong anonymity by modifying two new transaction types: *mint* and *spend*. The *Mint* allows users to exchange some quantities of bitcoins to mint a new zerocoin, which consists of a commitment number *cm* and a random serial number *sr*. The *Spend* represents a successful transmission from a corresponding amount of bitcoins to the destination address, when the commitment is correctly verified via zero-knowledge proof and the serial number has not been spent previously. Technically, Zerocoin employs the Pedersen commitments over a prime field $\mathbb{Z}_p$ as $cm = g^{sr}h^r$. In order to efficiently prove the commitment belongs to the specific set, it employs a one-way accumulator based on *Strong-RSA* [39] to reduce the size from $O(N)$ to $Olog(N)$. As the development of Zerocoin, Eli et al. [31] has extended the system into a full-fledged decentralized electronic currency called *Zerocash*, which formally captures the functionality and the security guarantees. The modified scheme reduces both the size of transactions and verifications, and allows for anonymous transactions in variable amounts when compared to its origin. Fundamentally, it uses the technical zero knowledge proof called *zero-knowledge Succinct Non-interactive ARguments of Knowledge(zk-SNARKs)*. The advanced system achieves implementation with strong anonymity guarantees under the coalition of academic departments and industrial organizations.

However, the Zerocoin system confronts several drawbacks on functionalities and efficiency. For functionality, Zerocoin defines its coins as a fixed denomination, which only supports certain divided values other than arbitrary values in real transactions. The fixed denomination makes hidden information (like identities, address) feasible, but not contains the area of amounts. Our scheme can arbitrarily encrypt amounts in frequent transactions, and the homomorphic property makes ciphertexts operated in practical way. For efficiency, Zerocoin employs the double-discrete-logarithm zero-knowledge proof, which leads to large proof sizes and verification times. Our scheme embeds the Paillier cryptosystem and the commitment system into original Bitcoin system, the fundamental computation only refers to modular operation which is theoretically simpler than Zerocoin. The modified Zerocash system has partially improved the above drawbacks, but still in the same foundation. Our scheme provides a new way to deal with it.

Our scheme employs the Paillier cryptosystem as a homomorphically encrypted tool and uses two kinds of the commitment proofs as zero knowledge proof protocol. The underlying idea of the above schemes adheres to a common design principle: **Encryption/Confusion→Zero-knowledge/Commitment proof**. The privacy leaked in the Bitcoin system requires a cryptographic tool to hide it, then the hidden information results in a transmission of trust which demands a zero-knowledge proof. Different schemes select different tools but follow the same principle.

5. Conclusion

We have proposed an improved delicately anonymous transaction scheme on top of the Bitcoin system by using the Paillier cryptosystem to improve the privacy. Our scheme can hide the amounts by encrypting the plaintext into the ciphertext. Furthermore, we created the *dumb account* as a bridge between the system and the traders, to realize the verification consisting of the positiveness check of the encrypted amounts and the equality check of the Input&Output sums. Our scheme has a high security level which can prevent both active and passive attacks. Analyses have shown that our scheme is functional, effective and practical for applications. This brand new scheme provides an inspiring way to achieve delicate confidentiality of the transactions in many applications.

Acknowledgments

This paper is supported by the Natural Science Foundation of China through projects 61772538, 61672083, 61370190, 61532021, 61472429, and 61402029, by the Beijing Natural Science Foundation through project 4132056.

References

[1] D. Chaum, Blind signatures for untraceable payments, in: Advances in Cryptology: Proceedings of CRYPTO '82, Plenum, 1982, pp. 199–203.
[2] R.L. Rivest, Peppercoin micropayments, in: Financial Cryptography,8th International Conference, FC 2004, Key West, FL, USA, February 9–12, 2004 Revised Papers, 2004, pp. 2–8. http://dx.doi.org/10.1007/978-3-540-27809-2_2.
[3] B. Schoenmakers, Security aspects of the ecash payment system, Lecture Notes in Comput. Sci. (1998) 338–352. http://dx.doi.org/10.1007/3-540-49248-8_16.
[4] S. Nakamoto, Blind signatures for untraceable payments, 2008. https://bitcoin.org/bitcoin.
[5] Wiki.Transactions. https://en.bitcoin.it/wiki/Transactions.
[6] C. Dwork, M. Naor, Pricing via processing or combatting junk mail, in: International Cryptology Conference, 740, 1992, pp. 139–147.
[7] Litecoin. https://www.litecoin.com.
[8] I. Miers, C. Garman, M. Green, A.D. Rubin, Zerocoin: Anonymous distributed e-cash from bitcoin, 2013, pp. 397–411.
[9] S. King, Primecoin: Cryptocurrency with prime number proof-of-work, 2013. http://primecoin.io/bin/primecoin-paper.pdf.
[10] D.M. Goldschlag, S.G. Stubblebine, Publicly verifiable lotteries: Applications of delaying functions, Lecture Notes in Comput. Sci. (1998) 214–226.
[11] R.L. Rivest, A. Shamir, Payword and micromint: Two simple micropayment schemes, 1996, pp. 69–87.
[12] M. Pease, R.E. Shostak, L. Lamport, Reaching agreement in the presence of faults, J. ACM 27 (2) (1980) 228–234.
[13] L. Lamport, R.E. Shostak, M.C. Pease, The byzantine generals problem, ACM Trans. Program. Lang. Syst. 4 (3) (1982) 382–401.
[14] J.J.L. Andre Miller, Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin, University of Central Florida. Tech Report, CS-TR-14-01.
[15] A.T. Tomas Sander, Shmaauditable.anonymous electronic cash, in: CRYPTO.
[16] B. Preneel, The first 30 years of cryptographic hash functions and the nist sha-3 competition, Lecture Notes in Comput. Sci. 5985 (2010) 1–14.
[17] F. 180-1. Secure hash standard, 1996.
[18] X. Wang, Y.L. Yin, H. Yu, Finding collisions in the full SHA-1, 2005, pp. 17–36. http://dx.doi.org/10.1007/11535218_2.
[19] Wiki.Elliptic curve digital signature algorithm. https://en.bitcoin.it/wiki/EllipticCurveDigitalSignatureAlgorithm.
[20] J. Bonneau, A.J. Miller, J. Clark, A. Narayanan, J.A. Kroll, E.W. Felten, Sok: Research perspectives and challenges for bitcoin and cryptocurrencies, 2015, pp. 104–121.
[21] J.A. Garay, A. Kiayias, N. Leonardos, The bitcoin backbone protocol: Analysis and applications, 2015, pp. 281–310.
[22] Q. Wu, X. Zhou, B. Qin, J. Hu, J. Liu, Y. Ding, Secure joint bitcoin trading with partially blind fuzzy signatures, Soft Comput. (2015) 1–12.
[23] Wiki.Proof-of-Stake, https://en.wikipedia.org/wiki/Proof-of-stake.
[24] Bitshares. https://www.btsabc.org.
[25] Ethereum. https://www.ethereum.org.
[26] Smart contract white paper. https://github.com/ethereum/wiki/wiki/White-Paper.
[27] A. Back, Bitcointalk.2013.bitcoins with homomorphic value, 2014. https://bitcointalk.org/index.php?topic=305791.0.
[28] G. Maxwell, Coinjoin: Bitcoin privacy for the real world, 2013. https://bitcointalk.org/index.php?topic=279249.0.2013.
[29] G. Maxwell, Confidential transactions, 2015. https://people.xiph.org/~greg/confidential-values.txt.2015.
[30] S. Noether, Ring signature confidential transcation for monero, 2015. http://eprint.iacr.org/2015/1098.
[31] E.B. Sasson, A. Chiesa, C. Garman, Zerocash: Decentralized anonymous payments from bitcoin, in: IEEE Symposium on Security and Privacy. http://dx.doi.org/10.1109/SP.2014.36.
[32] C. Decker, R. Wattenhofer, Information propagation in the bitcoin network, 2013. pp. 1–10.
[33] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in: Theory and Application of Cryptographic Techniques 1592, 1999, pp. 223–238.
[34] F. Boudot, Efficient proofs that a committed number lies in an interval, in: Theory and Application of Cryptographic Techniques, 1807, 2000, pp. 431–444.
[35] E. Fujisaki, T. Okamoto, Statistical zero knowledge protocols to prove modular polynomial relations, in: International Cryptology Conference, 1294, 1997, pp. 16–30.

[36] E.F. Brickell, D. Chaum, I. Damgard, J.V.D. Graaf, Gradual and verifiable release of a secret.
[37] A. Chan, Y. Frankel, Y. Tsiounis, Easy come easy go divisible cash, in: Theory and Application of Cryptographic Techniques, 1403, 1998, pp. 561–575.
[38] Q. Wu, et al., Simple proof that a committed number is in a specific interval, Chin. J. Electron. 32 (7) (2004). http://dx.doi.org/10.3321/j.issn:0372-2112.2004.07.004.
[39] J. Camenisch, A. Lysyanskaya, An efficient system for non-transferable anonymous credentials with optional anonymity revocation, in: Theory and Application of Cryptographic Techniques 2045, 2001, pp. 93–118.

Qin Wang received his Bachelor of Engineering degrees in Northwestern Polytechnical University in 2015. He is now pursuing a master's degree in information and communication engineering in Beijing University of Aeronautics and Astronautics. His research interests include distributed computing system security, biometric information security, block chain and cryptographic currencies.

Bo Qin received her Ph.D. degree in Cryptography from Xidian University in 2008 in China. Since then, she has been with Xi'an University of Technology (China) as a lecturer and with Universitat Rovira i Virgili (Catalonia) as a postdoctoral researcher. She is currently a lecturer in the Renmin University in China. Her research interests include pairing-based cryptography, data security and privacy, and VANET security. She has been a holder/co-holder of 5 China/Spain funded projects. She has authored over 60 publications and served in the program committee of several international conferences in information security.

Jiankun Hu received the Ph.D. degree in control engineering from the Harbin Institute of Technology, China, in 1993, and the master's degree in computer science and software engineering from Monash University, Australia, in 2000. He was a Research Fellow with the Delft University of Technology, The Netherlands, from 1997 to 1998, and Melbourne University, Australia, from 1998 to 1999. He has been with Ruhr University, Bochum, Germany. He is currently a full Professor and Research Director of the Cyber Security Laboratory with the School of Engineering and Information Technology, University of New South Wales, Australia. He received the prestigious German Alexander von Humboldt Fellowship from Ruhr University from 1995 to 1996. His main research interest is in the field of cyber security, including biometrics security, where he has published many papers in high-quality conferences and journals, including the IEEE Transactions on pattern analysis and machine intelligence. He has received seven Australian Research Council (ARC) Grants, and serves at the prestigious Panel of Mathematics, Information and Computing Sciences and the ARC Excellence in Research for Australia Evaluation Committee. He has served on the Editorial Board of up to seven international journals including IEEE Transactions on Information Forensics and Security, and served as the Security Symposium Chair of the IEEE ICC and the IEEE Globecom.

Fu Xiao Received the Ph.D. degree in Computer Science and Technology from Nanjing University of Science and Technology, Nanjing, China, in 2007. He is currently a Professor and Ph.D. supervisor with the School of Computer, Nanjing University of Posts and Telecommunications, Nanjing, China. His main research interests are wireless sensor networks and mobile computing. He has published over 30 papers in related international conferences and journals, including IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, ACM Transactions on Embedded Computing Systems, INFOCOM, IPCCC, ICC and so on. Dr. Xiao serves as TPC Member for several international Conference including the IEEE GLOBECOM, IFIP Networking. He is a member of the IEEE Computer Society and the Association for Computing Machinery.