

BLOCKCHAIN-BASED TRANSACTION METHOD AND APPARATUS, AND REMITTER DEVICE

TECHNICAL FIELD

[0001] Implementations of the present specification relate to the field of computer technologies, and particularly, to a blockchain-based transaction method and apparatus, and a remitter device.

BACKGROUND

[0002] The blockchain technology is a distributed database technology that protects data from being tampered with and forged by using cryptography and consensus mechanisms. With the development of computer and Internet technologies, thanks to advantages such as decentralization, openness and transparency, tamper-resistance, and trustiness, the blockchain technology is highly favored and is widely applied to many fields such as smart contracts, securities transactions, E-commerce, the Internet of Things, social communication, file storage, proof of existence, identity verification, and equity crowdfunding.

[0003] Currently, when the blockchain technology is applied to transaction scenarios, because transaction information needs to be sent to the blockchain for verification, implementation, and chaining-on, the transaction information is at risk of being leaked to a third party unrelated to the transaction.

[0004] There is a need of a technical solution for realizing privacy protection in a transaction process.

SUMMARY

[0005] The objective of implementations of the present specification is to provide a blockchain-based transaction method and apparatus, and a remitter device, to achieve privacy protection without interaction with a remittee device.

[0006] To achieve the above objective, an implementation of the present specification provides a blockchain-based transaction method, including: calculating a transaction amount commitment based on a commitment random number and a

transaction amount; obtaining a first commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remitter account and the transaction amount using a homomorphic encryption algorithm; obtaining a second commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remittee account and the transaction amount using the homomorphic encryption algorithm; and submitting transaction data to the blockchain, the transaction data including the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

[0007] To achieve the above objective, an implementation of the present specification provides a blockchain-based transaction apparatus, including: a calculation unit, configured to calculate a transaction amount commitment based on a commitment random number and a transaction amount; a first encryption unit, configured to obtain a first commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remitter account and the transaction amount using a homomorphic encryption algorithm; a second encryption unit, configured to obtain a second commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remittee account and the transaction amount using the homomorphic encryption algorithm; and a submission unit, configured to submit transaction data to the blockchain, the transaction data including the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

[0008] To achieve the above objective, an implementation of the present specification further provides a remitter device, including: a memory, configured to store computer instructions; and a processor, configured to execute the computer instructions to implement the method described in any of the implementations of the

present specification.

[0009] It can be understood from the technical solutions provided in the implementations of the present specification, in the implementations of the present specification, the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext can be obtained by using the commitment mechanism and the homomorphic encryption mechanism. The transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext can be submitted to the blockchain, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account. On the one hand, privacy protection can be realized for transaction information such as the transaction amount by using the commitment mechanism and the homomorphic encryption mechanism. On the other hand, the transaction can be smoothly completed in the blockchain without the remittee device by submitting the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext to the blockchain. Therefore, the implementation can realize privacy protection under a non-interactive condition.

BRIEF DESCRIPTION OF DRAWINGS

[0010] To describe the technical solutions in the implementations of the present specification or in the existing technology more clearly, the following outlines the accompanying drawings for illustrating such technical solutions. Clearly, the accompanying drawings outlined below are some implementations of the present specification and a person skilled in the art can derive other drawings from such accompanying drawings without creative efforts.

[0011] FIG. 1 is a schematic diagram illustrating blockchain-based transaction implementation according to an implementation of the present specification;

[0012] FIG. 2 is a flowchart illustrating a blockchain-based transaction method according to an implementation of the present specification;

[0013] FIG. 3 is a flowchart illustrating blockchain-based transaction implementation according to an implementation of the present specification;

[0014] FIG. 4 is a function structural diagram illustrating a blockchain-based transaction apparatus according to an implementation of the present specification; and

[0015] FIG. 5 is a function structural diagram illustrating a remitter device according to an implementation of the present specification.

5

DESCRIPTION OF IMPLEMENTATIONS

[0016] The following clearly and comprehensively describes the technical solutions in the implementations of the present specification with reference to the accompanying drawings in the implementations of the present specification. Clearly, the described implementations are merely some rather than all of the implementations of the present specification. All other implementations obtained by a person of ordinary skill in the art based on the implementations of the present specification without creative efforts shall fall within the protection scope of the present specification.

[0017] Referring to FIG. 1 and FIG. 2, an implementation of the present specification provides a blockchain-based transaction method.

[0018] In this implementation, the blockchain can be a distributed ledger that organizes multiple pieces of block data in a chain structure based on a time sequence, and ensures security, traceability, and tamper-resistance by using a cryptography algorithm. The blockchain can include a public blockchain, a consortium blockchain, a private blockchain, etc. The blockchain can be implemented based on a blockchain network. The blockchain network can include a peer-to-peer network (P2P network), etc. The blockchain network can include a plurality of blockchain nodes. Unified blockchain ledgers are maintained among blockchain nodes together.

[0019] In the implementations, the transaction method can be executed by a remitter device. The remitter device can join the blockchain network as a blockchain node. A remitter account can be logged into on the remitter device. The remitter account can be an account of a remitter in the blockchain. The remitter account can have a signature public-private key pair, and an encryption public-private key pair. The signature public-private key pair can include a signature public key and a signature private key that are associated. The signature private key can be used to sign transaction data to be submitted to the blockchain. The signature public key can be disclosed to other blockchain nodes in the blockchain network for the other

blockchain nodes to verify the signed transaction data. The encryption public-private key pair can include a homomorphic encryption public key and a homomorphic encryption private key that are associated to one another. The homomorphic encryption public key can be disclosed to other blockchain nodes in the blockchain network, so the other blockchain nodes can encrypt data (*e.g.*, a remitter balance, a remitter random number, and a commitment random number) by using a homomorphic encryption algorithm. The homomorphic encryption private key can be used to decrypt an encrypted ciphertext. The remitter account can also register a remitter balance commitment and a remitter random number ciphertext in the blockchain.

[0020] The remitter balance commitment can be calculated by the remitter device based on the remitter random number and the remitter balance. The remitter random number can be a random number corresponding to the remitter account and used for constructing the remitter balance commitment. The remitter balance can be a balance of any type of resource. The resource can include, for example, securities, coupons, property, virtual currency, and funds. Specifically, the remitter balance commitment can be implemented based on any type of commitment scheme with homomorphism, such as a Pedersen commitment mechanism. Taking the Pedersen commitment mechanism as an example, the remitter balance commitment can be calculated based on equation $PC(r_A, s_A) = g^{r_A} h^{s_A}$, where, $PC(r_A, s_A)$ is the remitter balance commitment; r_A is the remitter random number; s_A is the remitter balance; g and h are pre-determined parameters, respectively. Certainly, the remitter balance commitment can also be implemented by using a homomorphic encryption algorithm. That is, a ciphertext obtained by encrypting the remitter balance based on the homomorphic encryption algorithm is used as the remitter balance commitment. The homomorphic encryption algorithm will be described in detail subsequently. As such, on the one hand, direct registration of the remitter balance can be avoided by registering the remitter balance commitment in the blockchain, so the remitter balance can be hidden and kept secret. On the other hand, because the commitment mechanism can support a high-efficiency zero-knowledge proof, the remitter balance commitment can make a zero-knowledge proof (*e.g.*, a third zero-knowledge proof in the subsequent process) process involving the remitter balance in the subsequent process more efficient.

[0021] The remitter random number ciphertext can be obtained by encrypting the remitter random number based on the remitter balance by using the homomorphic encryption algorithm. Specifically, the remitter device can obtain the remitter random number ciphertext by encrypting the remitter random number based on the homomorphic encryption public key of the remitter account and the remitter balance using the homomorphic encryption algorithm. The homomorphic encryption algorithm can include, for example, the Okamoto-Uchiyama algorithm and the Boneh-Goh-Nissim algorithm. Taking the Okamoto-Uchiyama algorithm as an example, the remitter random number ciphertext can be calculated based on equation

$$OU_A(r_A) = u1^{r_A} v1^{s_A},$$

where $OU_A(r_A)$ is the remitter random number ciphertext; r_A is the remitter random number; each of $u1$ and $v1$ is a portion of the homomorphic encryption public key of the remitter account; s_A is the remitter balance. In the implementations, the remitter random number can be used as an encrypted plaintext, and the remitter balance can be used as a random number for homomorphic encryption, thereby enabling encryption of the remitter random number. On the one hand, the burden caused by locally keeping the remitter random number off the chain can be avoided by registering the remitter random number in the blockchain. On the other hand, direct registration of the remitter random number can be avoided by registering the remitter random number ciphertext in the blockchain, so the remitter random number can be hidden and kept secret.

[0022] It should be noted that, when it is necessary to obtain the remitter balance, the remitter device can obtain the remitter random number by decrypting the remitter random number ciphertext using a homomorphic encryption private key of the remitter account; and can calculate the remitter balance based on the remitter random number and the remitter balance commitment. The remitter device can calculate the remitter balance by using, for example, the Kangaroo method in the Pollard algorithm. Certainly, the Kangaroo method listed herein is merely an example, and in practice the remitter device can calculate the remitter balance in other ways. For example, the remitter random number ciphertext can be $OU_A(r_A) = u1^{r_A} v1^{s_A}$, and the remitter balance commitment can be $PC(r_A, s_A) = g^{r_A} h^{s_A}$. Then, the remitter device can obtain the remitter random number r_A by decrypting the remitter random number ciphertext $OU_A(r_A)$ using the homomorphic encryption private key of the remitter account; calculate h^{s_A} based on the remitter random

number $r - A$ and the remitter balance commitment $PC(r - A, s - A)$; and calculate the remitter balance $s - A$ based on h^{s-A} by using the Kangaroo method in the Pollard algorithm. In addition, when the remitter balance commitment is implemented by using the homomorphic encryption algorithm, the remitter device can obtain the remitter balance by decrypting the remitter balance commitment using the homomorphic encryption private key of the remitter account. Specifically, for example, the remitter balance commitment implemented by using the Okamoto-Uchiyama algorithm can be $PC(r - A, s - A) = u1^{s-A}v1^{r-A}$.

[0023] In the implementations, the transaction method can transfer the transaction amount from the remitter account to a remittee account. The remittee account can be an account of a remittee in the blockchain. The remittee account can be logged into on a remittee device. The remittee device can join the blockchain network as a blockchain node. The transaction amount can be negotiated between the remitter and the remittee. Similar to the remitter account, the remittee account can have a signature public-private key pair, and an encryption public-private key pair. The remittee account can also register a remittee balance commitment and a remittee random number ciphertext in the blockchain. The remittee balance commitment can be calculated by the remittee device based on the remittee random number and the remittee balance. The remittee random number ciphertext can be obtained by encrypting the remittee random number based on the remittee balance by using the homomorphic encryption algorithm.

[0024] The transaction method can include the following steps. It should be noted that although the present specification provides the method operation steps in the implementations or the flowcharts, more or fewer operation steps can be included based on conventional or non-creative efforts. In addition, a sequence of the steps listed in the implementation is merely one of numerous execution sequences of the steps, and does not represent a unique execution sequence. In actual execution of an apparatus or a product, execution can be performed based on the method sequence shown in the implementation or the accompanying drawing, or performed in parallel (e.g., a parallel processor or a multi-thread processing environment).

[0025] Step S10: Calculate a transaction amount commitment based on a commitment random number and a transaction amount.

[0026] In the implementations, the commitment random number can be generated

by the remitter device. The transaction amount can be negotiated between the remitter and the remittee. The transaction amount commitment can be implemented based on any type of commitment scheme with homomorphism, such as a Pedersen commitment mechanism. Taking the Pedersen commitment mechanism as an example, the transaction amount commitment can be calculated based on equation $PC(r, t) = g^r h^t$, where, $PC(r, t)$ is the transaction amount commitment; r is the commitment random number; t is the transaction amount; g and h are pre-determined parameters, respectively. As such, the transaction amount can be hidden and kept secret by using the transaction amount commitment.

[0027] Step S12: Obtain a first commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remitter account and the transaction amount using a homomorphic encryption algorithm.

[0028] In the implementations, the homomorphic encryption algorithm can include, for example, the Okamoto-Uchiyama algorithm and the Boneh-Goh-Nissim algorithm. Taking the Okamoto-Uchiyama algorithm as an example, the first commitment random number ciphertext can be calculated based on equation $OU_A(r) = u_1^r v_1^t$, where $OU_A(r)$ is the first commitment random number ciphertext; r is the commitment random number; each of u_1 and v_1 is a portion of the homomorphic cryptographic public key of the remitter account, respectively; t is the transaction amount. In the implementations, the commitment random number can be used as an encrypted plaintext, and the transaction amount can be used as a random number for homomorphic encryption, thereby enabling encryption of the commitment random number.

[0029] Step S14: Obtain a second commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remittee account and the transaction amount using the homomorphic encryption algorithm.

[0030] In the implementations, the homomorphic encryption algorithm can include, for example, the Okamoto-Uchiyama algorithm and the Boneh-Goh-Nissim algorithm. Taking the Okamoto-Uchiyama algorithm as an example, the second commitment random number ciphertext can be calculated based on equation $OU_B(r) = u_2^r v_2^t$, where $OU_B(r)$ is the second commitment random number

ciphertext; r is the commitment random number; each of u_2 and v_2 is a portion of the homomorphic cryptographic public key of the remittee account, respectively; t is the transaction amount. In the implementations, the commitment random number can be used as an encrypted plaintext, and the transaction amount can be used as a random number for homomorphic encryption, thereby enabling encryption of the commitment random number.

[0031] Step S16: Submit transaction data to the blockchain.

[0032] In the implementations, the transaction data can include the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext. The remitter device can submit the transaction data to the blockchain, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

[0033] Specifically, after the transaction data is submitted to the blockchain, a consensus blockchain node in the blockchain network can update the remitter balance commitment based on the transaction amount commitment; update the remitter random number ciphertext based on the first commitment random number ciphertext; update the remittee balance commitment based on the transaction amount commitment; and update the remittee random number ciphertext based on the second commitment random number ciphertext. The consensus blockchain node can be a blockchain node determined based on a consensus mechanism of the blockchain network. As such, the transaction data can be recorded into a blockchain ledger maintained by each blockchain node in the blockchain network, so as to avoid tampering.

[0034] An updated remitter balance commitment can be a quotient of the remitter balance commitment and the transaction amount commitment. Because the remitter balance commitment and the transaction amount commitment are implemented by using a homomorphic commitment mechanism, the transaction amount can be deducted from the remitter balance and the commitment random number can be deducted from the remitter random number. For example, the remitter balance commitment can be $PC(r, A, s, A) = g^{r-A} h^{s-A}$; the transaction amount commitment can be $PC(r, t) = g^r h^t$; the updated remitter balance commitment can be

$$PC(r - A - r, s - A - t) = PC(r - A, s - A) / PC(r, t) = g^{(r - A - r)} h^{(s - A - t)}.$$

[0035] An updated remitter random number ciphertext can be a quotient of the remitter random number ciphertext and the first commitment random number ciphertext. Because the remitter random number ciphertext and the first commitment random number ciphertext are calculated by using the homomorphic encryption algorithm, the transaction amount can be deducted from the remitter balance, and the commitment random number can be deducted from the remitter random number. For example, the remitter random number ciphertext can be $OU - A(r - A) = u1^{r-A} v1^{s-A}$; the first commitment random number ciphertext can be $OU - A(r) = u1^r v1^t$; the updated remitter random number ciphertext can be $OU - A(r - A - r) = OU - A(r - A) / OU - A(r) = u1^{r-A-r} v1^{s-A-t}$.

[0036] An updated remittee balance commitment can be a product of the remittee balance commitment and the transaction amount commitment. Because the remittee balance commitment and the transaction amount commitment are implemented by using the homomorphic commitment mechanism, the transaction amount can be added to the remittee balance and the commitment random number can be added to the remittee random number. For example, the remittee balance commitment can be $PC(r - B, s - B) = g^{r-B} h^{s-B}$; the transaction amount commitment can be $PC(r, t) = g^r h^t$; the updated remittee balance commitment can be $PC(r - B + r, s - B + t) = PC(r - B, s - B) PC(r, t) = g^{(r-B+r)} h^{(s-B+t)}$. $r - B$ is the remittee random number; $s - B$ is the remittee balance; g and h are pre-determined parameters, respectively.

[0037] An updated remittee random number ciphertext can be a product of the remittee random number ciphertext and the second commitment random number ciphertext. Because the remittee random number ciphertext and the second commitment random number ciphertext are calculated by using the homomorphic encryption algorithm, the transaction amount can be added to the remittee balance, and the commitment random number can be added to the remittee random number. For example, the remittee random number ciphertext can be $OU - B(r - B) = u2^{r-B} v2^{s-B}$; the second commitment random number ciphertext can be $OU - B(r) = u2^r v2^t$; the updated remittee random number ciphertext can be

$$OU - B(r - B + r) = OU - B(r - B)OU - B(r) = u 2^{r-B+r} v 2^{s-B+t}.$$

[0038] In an implementation, the remitter device can further generate a first zero-knowledge proof based on a zero-knowledge proof technique; and add the first zero-knowledge proof to the transaction data for the consensus blockchain node to verify that the transaction amount is not less than 0. The zero-knowledge proof technique can be implemented, for example, based on the zkSNARK scheme. Further, the zero-knowledge proof technique can include a range proof technique. The remitter device can generate a first range proof based on the range proof technique; and add the first range proof to the transaction data for the consensus blockchain node to verify that the transaction amount is not less than 0. The range proof technique can be implemented, for example, based on the Bulletproofs scheme or the Borromean ring signature scheme.

[0039] In an implementation, the remitter device can further generate a second zero-knowledge proof based on the zero-knowledge proof technique; and add the second zero-knowledge proof to the transaction data for the consensus blockchain node to verify that the transaction amount is not greater than the remitter balance. Further, the zero-knowledge proof technique can include the range proof technique. The remitter device can generate a second range proof based on the range proof technique; and add the second range proof to the transaction data for the consensus blockchain node to verify that the transaction amount is not greater than the remitter balance.

[0040] In an implementation, the remitter device can further generate a third zero-knowledge proof based on the zero-knowledge proof technique; and add the third zero-knowledge proof to the transaction data for the consensus blockchain node to verify that the commitment random number for calculating the transaction amount commitment, the commitment random number for calculating the first commitment random number ciphertext, and the commitment random number for calculating the second commitment random number ciphertext are consistent with one another, and verify that the transaction amount for calculating the transaction amount commitment, the transaction amount for calculating the first commitment random number ciphertext, and the transaction amount for calculating the second commitment random number ciphertext are consistent with one another.

[0041] For example, the third zero-knowledge proof can be (C, D, E, a, b).

$C = g^{r^*} h^{t^*}$; $D = u_1^{r^*} v_1^{t^*}$; $E = u_2^{r^*} v_2^{t^*}$; $x = Hash(C, D, E)$; $a = r^* + xr$; $b = t^* + xt$.
 r^* and t^* are random numbers generated by the remitter device and respectively correspond to the commitment random number r and the transaction amount t ; Hash represents a hash operation.

5 **[0042]** When verifying that all $g^a h^b = CT^x$, $u_1^a v_1^b = DM^x$, and
 $u_2^a v_2^b = EN^x$ are valid, the consensus blockchain node considers that the
 commitment random number for calculating the transaction amount commitment, the
 commitment random number for calculating the first commitment random number
 ciphertext, and the commitment random number for calculating the second
 10 commitment random number ciphertext are consistent with one another, and considers
 that the transaction amount for calculating the transaction amount commitment, the
 transaction amount for calculating the first commitment random number ciphertext,
 and the transaction amount for calculating the second commitment random number
 ciphertext are consistent with one another. T is the transaction amount commitment
 15 $PC(r, t)$; M is the first commitment random number ciphertext $OU - A(r)$; N is
 the second commitment random number ciphertext $OU - B(r)$.

[0043] In an implementation, the remitter device can calculate the first transaction
 amount commitment based on the first commitment random number and a first
 transaction amount; and calculate the second transaction amount commitment based
 20 on the second commitment random number and a second transaction amount. The first
 commitment random number and the second commitment random number can be
 generated by the remitter device. The first transaction amount can be a transaction
 amount to be transferred out of the remitter account. The second transaction amount
 can be a transaction amount to be transferred to the remittee account. As such, the
 25 transaction amount to be transferred out of the remitter account and the transaction
 amount to be transferred to the remittee account can be distinguished.

[0044] The remitter device can obtain the first commitment random number
 ciphertext by encrypting the first commitment random number based on the
 homomorphic encryption public key of the remitter account and the first transaction
 30 amount using the homomorphic encryption algorithm; obtain the second commitment
 random number ciphertext by encrypting the second commitment random number
 based on the homomorphic encryption public key of the remittee account and the
 second transaction amount using the homomorphic encryption algorithm; and submit

the transaction data to the blockchain. The transaction data can include the first transaction amount commitment, the second transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the first transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the second transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

[0045] Specifically, after the transaction data is submitted to the blockchain, the consensus blockchain node can update the remitter balance commitment based on the first transaction amount commitment; update the remitter random number ciphertext based on the first commitment random number ciphertext; update the remittee balance commitment based on the second transaction amount commitment; and update the remittee random number ciphertext based on the second commitment random number ciphertext.

[0046] An updated remitter balance commitment can be a quotient of the remitter balance commitment and the first transaction amount commitment; an updated remitter random number ciphertext can be a quotient of the remitter random number ciphertext and the first commitment random number ciphertext; an updated remittee balance commitment can be a product of the remittee balance commitment and the second transaction amount commitment; and an updated remittee random number ciphertext can be a product of the remittee random number ciphertext and the second commitment random number ciphertext.

[0047] Further, the remitter device can further generate a fourth zero-knowledge proof based on the zero-knowledge proof technique; and add the fourth zero-knowledge proof to the transaction data for the consensus blockchain node to verify that the first transaction amount for calculating the first transaction amount commitment and the second transaction amount for calculating the second transaction amount commitment are consistent with one another, so as to prevent the transaction amount transferred out of the remitter account from being inconsistent with the transaction amount transferred to the remittee account.

[0048] In an implementation, the remitter device can further obtain signature data by signing the transaction data using a signature private key of the remitter account; and add the signature data to the transaction data for the consensus blockchain node to perform signature verification.

[0049] In the implementations, the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext can be obtained by using the commitment mechanism and the homomorphic encryption mechanism. The transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext can be submitted to the blockchain, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account. On the one hand, privacy protection can be realized for transaction information such as the transaction amount by using the commitment mechanism and the homomorphic encryption mechanism. On the other hand, the transaction can be smoothly completed in the blockchain without the remittee device by submitting the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext to the blockchain. Therefore, the implementation can realize privacy protection under a non-interactive condition.

[0050] Further, the remitter account can register the remitter balance commitment and the remitter random number ciphertext in the blockchain. The remittee account can register the remittee balance commitment and the remittee random number ciphertext in the blockchain. On the one hand, privacy protection can be realized for both the account balance and the transaction amount. On the other hand, by registering the remitter random number ciphertext and the remittee random number ciphertext on the blockchain, the burden caused by locally keeping the remitter random number and the remittee random number off the chain can be avoided, and the risk of loss can be avoided.

[0051] Refer to FIG. 1, FIG. 2 and FIG. 3. For ease of understanding, an example scenario of an implementation of the present specification is described below.

[0052] In this implementation, assume that the remitter account can be Account_A. The remitter account Account_A can have a signature public-private key pair and an encryption public-private key pair. The remitter account Account_A can also register a remitter balance commitment $PC(r_A, s_A) = g^{r_A} h^{s_A}$ and a remitter random number ciphertext $OU_A(r_A) = u1^{r_A} v1^{s_A}$. r_A is the remitter random number; s_A is the remitter balance; g and h are

pre-determined parameters; each of u_1 and v_1 is a portion of the homomorphic encryption public key of the remitter account.

[0053] Assume that the remittee account can be Account_B. The remittee account Account_B can have a signature public-private key pair and an encryption public-private key pair. The remittee account Account_B can also register a remittee balance commitment $PC(r_B, s_B) = g^{r_B} h^{s_B}$ and a remittee random number ciphertext $OU_B(r_B) = u_2^{r_B} v_2^{s_B}$. r_B is the remittee random number; s_B is the remittee balance; g and h are pre-determined parameters; each of u_2 and v_2 is a portion of the homomorphic encryption public key of the remittee account.

[0054] In the implementations, the example transaction of the scenario can implement the transfer of transaction amount t from the remitter account Account_A to the remittee account Account_B. Specifically, the remitter device can generate the commitment random number r ; calculate the transaction amount commitment $PC(r, t) = g^r h^t$ based on the commitment random number r and transaction amount t ; obtain the first commitment random number ciphertext $OU_A(r) = u_1^r v_1^t$ by encrypting the commitment random number r based on the homomorphic encryption public key of the remitter account and transaction amount t using the homomorphic encryption algorithm; obtain the second commitment random number ciphertext $OU_B(r) = u_2^r v_2^t$ by encrypting the commitment random number r based on the homomorphic encryption public key of the remittee account and transaction amount t using the homomorphic encryption algorithm.

[0055] In this implementation, the remitter device can generate a first zero-knowledge proof $RP1$, a second zero-knowledge proof $RP2$, and a third zero-knowledge proof $RP3$. The first zero-knowledge proof $RP1$ can be used to verify: $t \geq 0$. The second zero-knowledge proof $RP2$ can be used to verify: $s_A - t \geq 0$. The third zero-knowledge proof $RP3$ can be used to verify that the commitment random number r in $PC(r, t) = g^r h^t$, the commitment random number r in $OU_A(r) = u_1^r v_1^t$, and the commitment random number r in $OU_B(r) = u_2^r v_2^t$ are consistent with one another, and verify that transaction amount t in $PC(r, t) = g^r h^t$, transaction amount t in $OU_A(r) = u_1^r v_1^t$, and transaction amount t in $OU_B(r) = u_2^r v_2^t$ are consistent with one another.

[0056] In this implementation, the remitter device can obtain signature data SIG by signing $[PC(r, t), OU_A(r), OU_B(r), RP1, RP2, RP3]$ using a signature private key of the remitter account; and submit transaction data $[PC(r, t), OU_A(r), OU_B(r), RP1, RP2, RP3, SIG]$ to the blockchain.

[0057] In this implementation, the blockchain network can determine a consensus blockchain node based on the consensus mechanism. By using the anti-double-spending or anti-replay mechanism of the related technologies, the consensus blockchain node can verify whether the transaction has been executed. If the transaction has been executed, the consensus blockchain node can reject the transaction.

[0058] If the transaction has not been executed, the consensus blockchain node can verify whether the signature data SIG in the transaction data is correct. If the signature data SIG is incorrect, the consensus blockchain node can reject the transaction.

[0059] If the signature data SIG is correct, the consensus blockchain node can check the first zero-knowledge proof $RP1$ in the transaction data to verify whether $t \geq 0$ is satisfied. If not satisfied, the consensus blockchain node can reject the transaction.

[0060] If satisfied, the consensus blockchain node can check the second zero-knowledge proof $RP2$ in the transaction data to verify whether $s_A - t \geq 0$ is satisfied. If not satisfied, the consensus blockchain node can reject the transaction.

[0061] If satisfied, the consensus blockchain node can check the third zero-knowledge proof $RP3$ in the transaction data to verify whether the commitment random number r in $PC(r, t) = g^r h^t$, the commitment random number r in $OU_A(r) = u1^r v1^t$, and the commitment random number r in $OU_B(r) = u2^r v2^t$ are consistent with one another, and verify whether transaction amount t in $PC(r, t) = g^r h^t$, transaction amount t in $OU_A(r) = u1^r v1^t$, and transaction amount t in $OU_B(r) = u2^r v2^t$ are consistent with one another. If not satisfied, the consensus blockchain node can reject the transaction.

[0062] If satisfied, the consensus blockchain node can update the remitter balance commitment $PC(r_A, s_A)$, the remitter random number ciphertext $OU_A(r_A)$, the remittee balance commitment $PC(r_B, s_B)$, and the remittee

random number ciphertext $OU_B(r_B)$. Specifically, the updated remitter balance commitment can be

$$PC(r_A - r, s_A - t) = PC(r_A, s_A) / PC(r, t) = g^{(r_A - r)} h^{(s_A - t)} ; \text{ the updated}$$

remitter random number ciphertext can be

$$OU_A(r_A - r) = OU_A(r_A) / OU_A(r) = u1^{r_A - r} v1^{s_A - t} ; \text{ an updated remittee}$$

balance commitment can be

$$PC(r_B + r, s_B + t) = PC(r_B, s_B) PC(r, t) = g^{(r_B + r)} h^{(s_B + t)} ; \text{ an updated}$$

remittee random number ciphertext can be

$$OU_B(r_B + r) = OU_B(r_B) OU_B(r) = u2^{r_B + r} v2^{s_B + t} .$$

10 **[0063]** Referring to FIG. 4, an implementation of the present specification further provides a blockchain-based transaction apparatus. The transaction apparatus can include the following units: a calculation unit 20, configured to calculate a transaction amount commitment based on a commitment random number and a transaction amount; a first encryption unit 22, configured to obtain a first commitment random number ciphertext by encrypting the commitment random number based on a
15 homomorphic encryption public key of a remitter account and the transaction amount using a homomorphic encryption algorithm; a second encryption unit 24, configured to obtain a second commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a
20 remittee account and the transaction amount using the homomorphic encryption algorithm; and a submission unit 26, configured to submit transaction data to the blockchain, the transaction data including the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the transaction amount commitment and the first commitment
25 random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

[0064] Referring to FIG. 5, an implementation of the present specification further provides a remitter device. The remitter device can include a memory and a processor.

30 **[0065]** In this implementation, the memory can be implemented in any suitable way. For example, the memory can be a read-only memory, a mechanical hard disk, a solid-state drive, a USB flash drive, etc. The memory can be configured to store computer instructions.

[0066] In this implementation, the processor can be implemented in any suitable way. For example, the processor can be a microprocessor, a processor, or a computer readable medium storing computer readable program code (such as software or firmware) that can be executed by the microprocessor or the processor, a logic gate, a switch, an application-specific integrated circuit (ASIC), a programmable logic controller, or a built-in microcontroller. The processor can execute the computer instructions to perform the steps of: calculating a transaction amount commitment based on a commitment random number and a transaction amount; obtaining a first commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remitter account and the transaction amount using a homomorphic encryption algorithm; obtaining a second commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remittee account and the transaction amount using the homomorphic encryption algorithm; and submitting transaction data to the blockchain, the transaction data including the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

[0067] The implementations in the present specification are described in a progressive method. For the same or similar parts in the implementations, references can be made to each other. Each implementation focuses on a difference from other implementations. Especially, an apparatus implementation and a device implementation are basically similar to a method implementation, and therefore are described briefly. For related parts, refer to descriptions in the method implementation.

[0068] After reading the present specification, a person skilled in the art can think of any combination between some or all of the implementations enumerated in the present specification without creative efforts. These combinations are also within the scope disclosed and protected by the present specification.

[0069] In the 1990s, whether technology improvement is hardware improvement (for example, improvement of a circuit structure, such as a diode, a transistor, or a switch) or software improvement (improvement of a method procedure) can be clearly

distinguished. However, as technologies develop, the current improvement for many method procedures can be considered as a direct improvement of a hardware circuit structure. A designer usually programs an improved method procedure to a hardware circuit, to obtain a corresponding hardware circuit structure. Therefore, a method procedure can be improved by using a hardware entity module. For example, a programmable logic device (PLD) (for example, a field programmable gate array (FPGA)) is such an integrated circuit, and a logical function of the programmable logic device is determined by a user through device programming. A designer performs programming to "integrate" a digital system to a single PLD, without requiring a chip manufacturer to design and manufacture a dedicated integrated circuit chip 2. In addition, at present, instead of manually manufacturing an integrated chip, this type of programming is mostly implemented by using "logic compiler" software. The programming is similar to a software compiler used to develop and write a program. Original code needs to be written in a particular programming language for compilation. The language is referred to as a hardware description language (HDL). There are many HDLs, such as the Advanced Boolean Expression Language (ABEL), the Altera Hardware Description Language (AHDL), Confluence, the Cornell University Programming Language (CUPL), HDCal, the Java Hardware Description Language (JHDL), Lava, Lola, MyHDL, PALASM, and the Ruby Hardware Description Language (RHDL). The very-high-speed integrated circuit hardware description language (VHDL) and Verilog2 are most commonly used. A person skilled in the art should also understand that a hardware circuit that implements a logical method procedure can be readily obtained once the method procedure is logically programmed by using the several described hardware description languages and is programmed into an integrated circuit.

[0070] The system, apparatus, module, or unit illustrated in the implementations can be implemented by using a computer chip or an entity, or can be implemented by using a product having a certain function. A typical implementation device is a computer. A specific form of the computer can be a personal computer, a laptop computer, a cellular phone, a camera phone, an intelligent phone, a personal digital assistant, a media player, a navigation device, an email transceiver device, a game console, a tablet computer, a wearable device, or any combination thereof.

[0071] It can be understood from the descriptions of the implementations that a person skilled in the art can clearly understand that the present specification can be

implemented by using software and a necessary general hardware platform. Based on such an understanding, the technical solutions in the present specification essentially or the part contributing to the existing technology can be implemented in a form of a software product. The computer software product can be stored in a storage medium, such as a ROM/RAM, a magnetic disk, or an optical disc, and includes several instructions for instructing a computer device (which can be a personal computer, a server, a network device, etc.) to execute the method described in the implementations of the present specification or in some parts of the implementations of the present specification.

[0072] The present specification can be applied to many general-purpose or dedicated computer system environments or configurations, for example, a personal computer, a server computer, a handheld device or a portable device, a tablet device, a multi-processor system, a microprocessor-based system, a set-top box, a programmable consumption electronic device, a network PC, a minicomputer, a mainframe computer, and a distributed computing environment including any one of the previous systems or devices.

[0073] The present specification can be described in the general context of computer executable instructions executed by a computer, for example, a program module. Generally, the program module includes a routine, a program, an object, a component, a data structure, etc., executing a specific task or implementing a specific abstract data type. The present specification can also be practiced in distributed computing environments. In the distributed computing environments, tasks are performed by remote processing devices connected through a communications network. In a distributed computing environment, the program module can be located in both local and remote computer storage media including storage devices.

[0074] Although the present specification is described by using the implementations, a person of ordinary skill in the art knows that many variations of the present specification can be made without departing from the spirit of the present specification. It is expected that the appended claims include these variations without departing from the spirit of the present specification.

CLAIMS

What is claimed is:

1. A blockchain-based transaction method, comprising:

calculating a transaction amount commitment based on a commitment random
5 number and a transaction amount;

obtaining a first commitment random number ciphertext by encrypting the
commitment random number based on a homomorphic encryption public key of a
remitter account and the transaction amount using a homomorphic encryption
algorithm;

10 obtaining a second commitment random number ciphertext by encrypting the
commitment random number based on a homomorphic encryption public key of a
remittee account and the transaction amount using the homomorphic encryption
algorithm; and

submitting transaction data to a blockchain, the transaction data including the
15 transaction amount commitment, the first commitment random number ciphertext, and
the second commitment random number ciphertext, for the transaction amount
commitment and the first commitment random number ciphertext to be recorded into
the remitter account, and the transaction amount commitment and the second
commitment random number ciphertext to be recorded into the remittee account.

20

2. The method according to claim 1, wherein the remitter account registers a
remitter balance commitment and a remitter random number ciphertext in the
blockchain; the remitter balance commitment is calculated based on a remitter random
number and a remitter balance; the remitter random number ciphertext is obtained by
25 encrypting the remitter random number based on the remitter balance by using the
homomorphic encryption algorithm; and

wherein the remittee account registers a remittee balance commitment and a
remittee random number ciphertext in the blockchain; the remittee balance
commitment is calculated based on a remittee random number and a remittee balance;
30 the remittee random number ciphertext is obtained by encrypting the remittee random
number based on the remittee balance by using the homomorphic encryption
algorithm.

3. The method according to claim 2, wherein the transaction amount commitment and the first commitment random number ciphertext are recorded into the remitter account by:

5 updating the remitter balance commitment based on the transaction amount commitment, and updating the remitter random number ciphertext based on the first commitment random number ciphertext; and

wherein the transaction amount commitment and the second commitment random number ciphertext are recorded into the remittee account by:

10 updating the remittee balance commitment based on the transaction amount commitment, and updating the remittee random number ciphertext based on the second commitment random number ciphertext.

4. The method according to claim 3, wherein an updated remitter balance commitment is a quotient of the remitter balance commitment and the transaction amount commitment; and the updated remitter random number ciphertext is a quotient of the remitter random number ciphertext and the first commitment random number ciphertext; and

an updated remittee balance commitment is a product of the remittee balance commitment and the transaction amount commitment; and an updated remittee random number ciphertext is a product of the remittee random number ciphertext and the second commitment random number ciphertext.

5. The method according to claim 1, further comprising:

before the submitting the transaction data to the blockchain,

25 generating a first zero-knowledge proof, the first zero-knowledge proof configured to verify that the transaction amount is not less than 0;

wherein, correspondingly, the transaction data further includes the first zero-knowledge proof.

30 6. The method according to claim 1, further comprising:

before the submitting the transaction data to the blockchain,

generating a second zero-knowledge proof, the second zero-knowledge proof configured to verify that the transaction amount is not greater than the remitter balance;

wherein, correspondingly, the transaction data further includes the second zero-knowledge proof.

7. The method according to claim 1, further comprising:

5 before the submitting the transaction data to the blockchain,

 generating a third zero-knowledge proof, the third zero-knowledge proof configured to verify that the commitment random number for calculating the transaction amount commitment, the commitment random number for calculating the first commitment random number ciphertext, and the commitment random number for
10 calculating the second commitment random number ciphertext are consistent with one another, and to verify that the transaction amount for calculating the transaction amount commitment, the transaction amount for calculating the first commitment random number ciphertext, and the transaction amount for calculating the second commitment random number ciphertext are consistent with one another;

15 wherein, correspondingly, the transaction data further includes the third zero-knowledge proof.

8. The method according to claim 1, further comprising:

 before the submitting the transaction data to the blockchain,

20 obtaining a signature data by signing the transaction data using a signature private key of the remitter account;

 wherein, correspondingly, the transaction data further includes the signature data.

25 9. The method according to claim 1, wherein the calculating the transaction amount commitment based on the commitment random number and the transaction amount includes:

 calculating a first transaction amount commitment based on a first commitment random number and a first transaction amount, and calculating a second
30 transaction amount commitment based on a second commitment random number and a second transaction amount;

 wherein the obtaining the first commitment random number ciphertext by encrypting the commitment random number based on the homomorphic encryption public key of the remitter account and the transaction amount using the homomorphic

encryption algorithm includes:

obtaining the first commitment random number ciphertext by encrypting the first commitment random number based on the homomorphic encryption public key of the remitter account and the first transaction amount using the homomorphic encryption algorithm; and

wherein the obtaining the second commitment random number ciphertext by encrypting the commitment random number based on the homomorphic encryption public key of the remittee account and the transaction amount using the homomorphic encryption algorithm includes:

obtaining the second commitment random number ciphertext by encrypting the second commitment random number based on the homomorphic encryption public key of the remittee account and the second transaction amount using the homomorphic encryption algorithm; and

wherein, correspondingly, the transaction data includes the first transaction amount commitment, the second transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the first transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the second transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

10. The method according to claim 9, further comprising:

before the submitting the transaction data to the blockchain,

generating a fourth zero-knowledge proof, the fourth zero-knowledge proof configured to verify that a first transaction amount for calculating the first transaction amount commitment and a second transaction amount for calculating the second transaction amount commitment are consistent with one another;

wherein, correspondingly, the transaction data further includes the fourth zero-knowledge proof.

11. A blockchain-based transaction apparatus, comprising:

a calculation unit, configured to calculate a transaction amount commitment based on a commitment random number and a transaction amount;

a first encryption unit, configured to obtain a first commitment random number

ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remitter account and the transaction amount using a homomorphic encryption algorithm;

5 a second encryption unit, configured to obtain a second commitment random number ciphertext by encrypting the commitment random number based on a homomorphic encryption public key of a remittee account and the transaction amount using the homomorphic encryption algorithm; and

10 a submission unit, configured to submit transaction data to the blockchain, the transaction data including the transaction amount commitment, the first commitment random number ciphertext, and the second commitment random number ciphertext, for the transaction amount commitment and the first commitment random number ciphertext to be recorded into the remitter account, and the transaction amount commitment and the second commitment random number ciphertext to be recorded into the remittee account.

15

12. A remitter device, comprising:

a memory, configured to store computer instructions; and

a processor, configured to execute the computer instructions to implement the method according to any one of claims 1 to 10.