scientific reports

Check for updates

OPEN

Quantum blockchain based on asymmetric quantum encryption and a stake vote consensus algorithm

Wusheng Wang, Yang Yu & Lingjie Du✉

As emerging next-generation information technologies, blockchains have unique advantages in information transparency and transaction security. They have attracted great attentions in social and financial fields. However, the rapid development of quantum computation and the impending realization of quantum supremacy have had significant impacts on the advantages of traditional blockchain based on traditional cryptography. Here, we propose a blockchain algorithm based on asymmetric quantum encryption and a stake vote consensus algorithm. The algorithm combines a consensus algorithm based on the delegated proof of stake with node behaviour and Borda count (DPoSB) and quantum digital signature technology based on quantum state computational distinguishability with a fully flipped permutation ($QSCD_{ff}$) problem. DPoSB is used to generate blocks by voting, while the quantum signature applies quantum one-way functions to guarantee the security of transactions. The analysis shows that this combination offers better protection than other existing quantum-resistant blockchains. The combination can effectively resist the threat of quantum computation on blockchain technology and provide a new platform to ensure the security of blockchain.

The concept[1] of blockchain technology was first introduced by Satoshi Nakamoto in 2008. Blockchain is a decentralized block of data linked in a chronological chain network to provide a distributed shared ledger and database. For example, in the first blockchain system, i.e. Bitcoin, each block contains two parts, namely, the block header and block body. The block header contains the hash value of the current block, the hash value of the previous block, the timestamp, and information about the Merkel tree; the block body contains the transaction information and the corresponding digital signature. One advantage of the blockchain is the usage of a distributed network, which provides the transparency and security of transaction information. After more than ten years of rapid development, this technology is not limited to Bitcoin and other cryptocurrencies but also attracts intense attention from multidisciplinary areas, such as finance, energy, medical care, and government affairs.

At the core of blockchain technologies, the most important aspects are consensus algorithms and digital signatures. Consensus algorithms can be used to generate blocks, while digital signatures can secure transaction information. For example, the consensus algorithm used in the Bitcoin network is proof of work (PoW)[1], which allows every miner to compete through computing power based on a hash algorithm. The miner with higher hash power tends to have larger probabilities to find the correct hash solution, and the first miner that finds the correct hash value will generate a new block. In addition, there are other consensus algorithms such as proof of stack (PoS)[2], delegated proof of stack (DPoS)[3], and delegated proof of stake with node's behaviour and Borda count(DPoSB)[4]. They do not rely on computing power and thus could lower the power consumption. There is also a Byzantine algorithm[5] that achieves consensus in communication in the presence of malicious nodes.

Digital signatures are an essential application of public-key cryptography. Encryption methods commonly used in the digital signatures of a classical blockchain are Rivest-Shamir-Adleman (RSA)[6] and elliptic curve cryptography (ECC)[7]. These well-developed encryption algorithms are too complex for classical computers to crack, ensuring the security of the digital signatures. However, Shor and others have found that a quantum algorithm can effectively solve the integer decomposition problem and the discrete logarithmic problem[8], which are the critical parts of the encryption methods. In this case, the security of blockchain technology based on the digital signatures is under the threat of quantum computation.

School of Physics and National Laboratory of Solid State Microstructures, Nanjing University, Nanjing 210093, China. ✉email: ljdu@nju.edu.cn

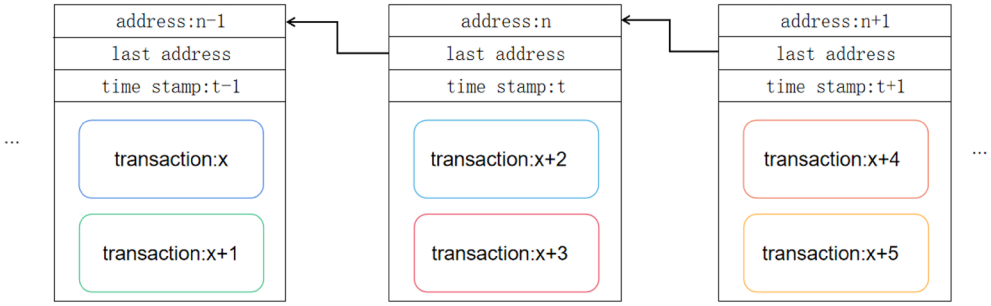

Figure 1. The data structure of the quantum blockchain. The block header contains the address of the current block, the address of the previous block and the timestamp. The block body contains the transaction that has passed through the quantum signature verification process. The arrows between two blocks indicate that we can find one block according to its next block.

Several physical systems have been developed to realize quantum computation. Quantum supremacy was demonstrated on a programmable superconducting quantum processor with 53 qubits by Google[9]. Pogorelov et al.[10] performed 50-qubit ion trap quantum computing. Moreover, Zhong et al.[11] demonstrated a 76-qubit quantum computer with photons for boson sampling and a programmable quantum nanophotonic chip with many photons[12].

Therefore, it has become urgent to develop new methods to protect against the threat of quantum computing. One effective approach is to develop quantum cryptography techniques based on the unique nature of quantum physics. For example, the quantum signature technology based on quantum state computational distinguishability with fully flipped permutations ($QSCD_{ff}$) problem, utilizing the complexity of $QSCD_{ff}$ problem for quantum computation, can guarantee the security of the signature process. In addition, there are also quantum key distribution (QKD) techniques used in quantum information, such as the most famous BB84 protocol[13]. These techniques help to improve security in communication processes even in the presence of quantum computation.

In this case, these algorithms can be involved in blockchain technologies, which further improve system securities. Several attempts have been made. For example, quantum key distribution (QKD) techniques, such as the most famous BB84 protocol[13], used in quantum information have been applied to blockchains[14]; quantum entanglement in time has been used to produce blocks[15], which is combined with quantum signature algorithms[16]. However, quantum signatures are not used in the QKD blockchain algorithm; a blockchain generated by the use of entanglement in time cannot trace back the transaction information, and thus the improvement in the overall security of the blockchain is poor.

To guarantee blockchain network security under quantum supremacy, we propose a quantum blockchain method that combines the DPoSB consensus algorithm[4] and quantum signatures established with quantum signature technology based on quantum state computational distinguishability with a fully flipped permutation ($QSCD_{ff}$) problem[17]. The former is developed from DPoS, which keeps the voting system and considers the influence of malicious behaviours in votes to improve security when malicious nodes are in a blockchain system. A quantum signature method using a quantum asymmetric cryptography approach is a signature method designed based on the complexity of the $QSCD_{ff}$ problem for quantum computation to guarantee the security of the signature process. Here we combine them together. The blockchain generates blocks by DPoSB and signs transactions by a quantum one-way function[18] based on the $QSCD_{ff}$ problem. Mining here is not necessary to make great savings on computing resources, which greatly saves computing resources and increases the speed of block generation. Different from other quantum signature methods[14,15,16], this method is not constrained[19,20] by probabilities and does not require a large number of one-time pads, which thus saves substantial communication overheads. Discussions about security models and quantum information-theoretical security are introduced in the security analysis. It can be found that our blockchain is secure even in the malicious adversary model. Our results show that this signature method in quantum blockchain is more secure than other quantum signatures. In this paper, the data structure of blockchain network is introduced in "Data structure of the blockchain " and our quantum blockchain section algorithm is analyzed in "Quantum blockchain algorithm" section. Then, the security of the blockchain algorithm is analyzed in "Security analysis of the blockchain" section, and the blockchain algorithm is compared with other existing quantum blockchains in "Comparison with other quantum blockchain signature methods" section. The conclusion is given at the end.

Data structure of the blockchain

A block acting as a unit in our blockchain system is constructed by a block header and a block body, as shown in Fig. 1. The information in the block header contains the address of the current block, the address of the previous block and the timestamp. The block body contains the transaction information that has passed through the quantum signature verification process. Due to the vital point of DPoSB, blockchain nodes do not need to participate in mining; namely, there is no computing force competition; thus, the hash value in the block is not necessary and can be replaced with the explicit address. We can begin from the block in the end to find the desired information according to the block addresses.

Quantum blockchain algorithm

In the blockchain, the signer generates the transaction and then uses a private key to sign, and the receiver authenticates the transaction by using the signer's public key to ensure transaction security in the aspect of cryptography.

First, our quantum blockchain network contains N nodes, and n (N>2n) witness nodes are elected to generate blocks in turn by DPoSB[13]. Then the nodes sign transactions through a quantum one-way function based on the $QSCD_{ff}$ problem. The witness nodes verify the transactions signed by the nodes and package the transactions into blockchain network if it passes through the verification process.

Blocks created by DPoSB. One key characteristic of DPoSB is voting, which is developed from DPoS. By the application of voting, the computing source originally used for mining can be largely saved. In voting, the n nodes with the highest votes are elected as the witness nodes responsible to generate blocks in turn. Let us assume that there are N nodes in a blockchain system. First, 2n (N>2n) candidate nodes are elected by voting, and then n witness nodes among the candidate nodes are elected. However, sometimes there are some malicious nodes appearing in the system, which hinder the generation of blocks.

There are four types of malicious behaviours denoted by r. Each r is distributed by a weight Q_r and the maximum threshold T_r is the largest number of times the behaviour r is accepted in the system. Below are the types of r,

r=1(fp): This indicates that the failure of transaction package, where $Q_1 = 0.4$ and T_1=Max1.

r=2 (fv): This indicates that the failure of block check, where $Q_2 = 0.3$ and T_2=Max2.

r=3 (bn): The failure of node communication, where $Q_3 = 0.2$ and T_3=Max3.

r=4 (other): Other types of malicious behaviour, where $Q_4 = 0.1$ and T_4=Max4.

Then DPoSB introduces malicious behaviour punishment calculation in the algorithm to address this issue and the mechanism of the Borda score to fairly select the witness nodes. We calculate the malicious behaviour weight ratio N_i^{Bw} for the i th node:

$$N_i^{Bw} = \sum_{r=1}^{B} \left(\frac{t_{ir}}{T_r} \times Q_r \right), (0 \le t_{ir} \le M_r, 0 \le i \le N, 0 \le r \le 4),$$

where t_{ir} represents the number of times the behaviour r is performed by the i th node makes.

The valid vote to define the i th node is:

$$V_i = \left(\sum_{j}^{N} P_j^{(t)} \right) \times \left(1 - N_i^{Bw}\right), \left(0 \le i \le N, 0 \le j \le N, 0 \le t\right),$$

where $P_j^{(t)}$ indicates the number of votes by j th node for i th node in round t of block generation (all participants produce a block once as the end of one round).

Then, we sort the valid votes for all nodes, and 2n nodes with the highest votes are elected as the candidate nodes.

The next step is to select n witness nodes from these candidate nodes. We construct the preference matrix:

$$\begin{bmatrix} r_{11}^k & r_{12}^k & \cdots & r_{1m}^k \\ r_{21}^k & r_{22}^k & \cdots & r_{2m}^k \\ \cdots & \cdots & \cdots & \cdots \\ r_{m1}^k & r_{m2}^k & \cdots & r_{mm}^k \end{bmatrix},$$

where $r_{ij}^k = \begin{cases} 1, \text{voter k prefers } x_i \succ x_j \\ 0, \text{voter k does not prefer } x_i \succ x_j \end{cases}$.

Then we have the k th node's preference value for the i th candidate node: $r_i^k = \sum_{j=1}^{N} r_{ij}^k$ and obtain the Borda score matrix:

$$\begin{bmatrix} r_1^1 & r_1^2 & \cdots & r_1^N \\ r_2^1 & r_2^2 & \cdots & r_2^N \\ \cdots & \cdots & \cdots & \cdots \\ r_C^1 & r_C^2 & \cdots & r_C^N \end{bmatrix}.$$

We calculate the cumulative Borda scores for each candidate node: $r_i = \sum_{k=1}^{N} r_i^k$.

The Borda scores are sorted for all candidate nodes, and the n candidate nodes with the highest scores are elected as the witness nodes.

The witness nodes can generate blocks in turn, as shown in Fig. 2.

Transaction signing and verification process. Then the nodes sign transactions through a quantum one-way function based on the quantum state computational distinguishability with fully flipped permutations $QSCD_{ff}$problem.

In quantum algorithms, quantum gate operations [21] can be performed on qubits, which include Hadamard (H), qubit flip (X), phase flip (Z) operations. The quantum state of a single qubit can be represented as $|\varphi\rangle = \sin\theta|0\rangle + \cos\theta e^{ia}|1\rangle$, where $|0\rangle$ and $|1\rangle$ are the counterparts of 0 and 1 in the classical computation. A

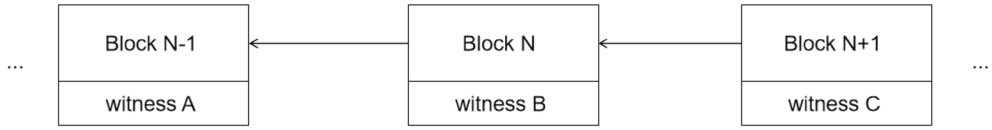

Figure 2. Witness nodes generate block in turn, and every witness node generates one block in one round.

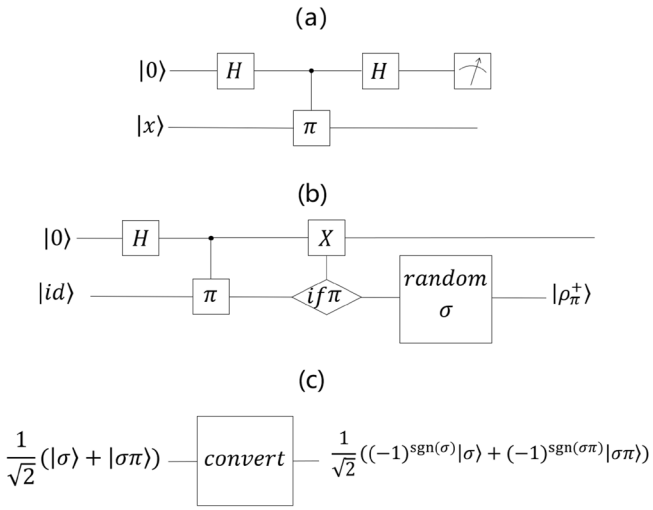

Figure 3. (**a**) The quantum circuit of the distinguishing algorithm, where H represents the Hadamard operations $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle$ and π represents the π operation $\pi|\sigma\rangle = |\sigma\pi\rangle$. We perform the quantum circuit from left to right. (**b**) The quantum circuit of the $\rho_\pi^+(n)$ generation algorithm, where "if π" means "if we read $|\pi\rangle$ from this register" and "random σ" means "perform a random permutation σ on this register". (**c**) The quantum circuit of the conversion algorithm, where we perform the following "convert" operation: $convert[\frac{1}{\sqrt{2}}(|\sigma\rangle + |\sigma\pi\rangle)] = \frac{1}{\sqrt{2}}\left((-1)^{sgn(\sigma)}|\sigma\rangle + (-1)^{sgn(\sigma\pi)}|\sigma\pi\rangle\right)$.

quantum gate operation can be represented as performing a unitary operator U on the quantum state, $U|\varphi\rangle$, to produce a target quantum state.

In the quantum algorithm, signing and verification processes are necessary to ensure a transaction. Here, we use the quantum one-way function based on the $QSCD_{ff}$ problem to finish the signing process.

A brief introduction to the $QSCD_{ff}$ *question.* We define $N_*=\{n \in N, n \text{ is even and } n/2 \text{ is odd}\}$. For each $n \in N_*$, S_n is used to represent a symmetric group of degree n. Then we use $\kappa_n=\{\pi \in S_n: \pi^2=id \text{ and } \forall i \in \{1,2,\cdots,n\}\ [\pi(i) \neq i]$, where id represents all the identity permutations. Each π can be represented as an odd permutation that is the product of $n/2$ disjoint transposition [22].

Then we have $|\kappa_n| = \frac{n!}{\left(\sqrt{2}\right)^n}$ and the following definition:

For each $\pi \in \kappa_n$, there are quantum states $\rho_\pi^+(n)$ and $\rho_\pi^-(n)$:

$$\rho_\pi^+(n) = \frac{1}{2n!}\sum\nolimits_{\sigma \in S_n}(|\sigma\rangle + |\sigma\pi\rangle)(\langle\sigma| + \langle\sigma\pi|),$$

$$\rho_\pi^-(n) = \frac{1}{2n!}\sum_{\sigma \in S_n}(|\sigma\rangle - |\sigma\pi\rangle)(\langle\sigma - \langle\sigma\pi).$$

For a symmetric group of degree n, each group element can be represented as an arrangement with n elements, such as a group element $(1,2,3)$ in S_3, which can be represented as quantum states: $|1\rangle|10\rangle|0\rangle$.

The $QSCD_{ff}$ problem is to distinguish the following two quantum states for each $n \in N_*$:$\rho_\pi^+(n)^{\otimes P(n)}$, $\rho_\pi^-(n)^{\otimes P(n)}$, where the $P(n)$ represents a polynomial.

Ref.[22] has proven that if $\pi \in \kappa_n$ is random and unknown there is no quantum algorithm that can solve the $QSCD_{ff}$ problem with non-negligible advantage. However, this problem can be quickly solved with the solution of π so that π would serve as a trapdoor in the quantum signatures.

A distinguishing algorithm for the $QSCD_{ff}$ problem. *Step 1* The quantum circuit used here is shown in Fig. 3a. For a quantum state x, $x \in \{\rho_\pi^+(n), \rho_\pi^-(n)\}$, we prepare the initial state $|0\rangle|x\rangle$. $|0\rangle$ is input into the first register

(a device used to preserve one or more quantum states) of the quantum circuit, and $|x\rangle$ is input into the second register. The Hadamard operation (H) is performed on $|0\rangle$, to obtain:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle, H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle.$$

Step 2 The C_π operation is performed on the second register controlled by the first register to obtain:

$$C_\pi|0\rangle|\sigma\rangle = |0\rangle|\sigma\rangle, C_\pi|1\rangle|\sigma\rangle = |1\rangle|\sigma\pi\rangle.$$

Step 3 The H operations is performed on the first register.

Step 4 The Z measurement is performed on the first register, and $x = \rho_\pi^+(n)$ if $|0\rangle$ is obtained, otherwise, $x = \rho_\pi^-(n)$.

$\rho_\pi^+(n)$ generating algorithm. $\rho_\pi^+(n)$ can be generated by the following steps. The quantum circuit is shown in Fig. 3b.

Step 1 We prepare the quantum state $|0\rangle|id\rangle$, input $|0\rangle$ into the first register, and input $|id\rangle$ into the second register. Then, we perform the H operation on the first register, and obtain $|+\rangle$ and the $|id\rangle$.

Step 2 The C_π operation is performed on the second register and controlled by the first register.

Step 3 If the second register reads $|\pi\rangle$, we perform a qubit flip operation (X) [21] on the first register.

Step 4 A uniformly random permutation σ is performed on the second register.

Step 5 The final state of the second register is output.

Converting algorithm. The symbol function $sgn(\cdot)$ on the symmetric group S_n is as follows:

If σ is an even permutation, $sgn(\sigma) = 0$; if σ is an odd permutation, $sgn(\sigma) = 1$.

We can convert $\rho_\pi^+(n)$ to $\rho_\pi^-(n)$ by the following operation:

$$\frac{1}{\sqrt{2}}(|\sigma\rangle + |\sigma\pi\rangle) \to \frac{1}{\sqrt{2}}\left((-1)^{sgn(\sigma)}|\sigma\rangle + (-1)^{sgn(\sigma\pi)}|\sigma\pi\rangle\right)$$

(even permutation × odd permutation = odd permutation, odd permutation × odd permutation = even permutation; $|\pi\rangle$ is an odd permutation.).

The quantum circuit is shown in Fig. 3c.

Signing transaction process. Below we use an example to show the detailed processes. Alice serves as a signer and Bob as a verifier. Jack acts as the private key generator (PKG), which is a trusted node in the blockchain system, and never exposes the signer's private key or imitates the signer to sign messages. Alice is ready to send a transaction message that she encodes as a bit string $TA(m_1, m_2, \cdots, m_n)$, $m_i \in \{0, 1\}$. The transaction can be signed by following steps [17].

Key generation phase. *Step 1* Alice randomly selects an odd permutation $\pi \in \kappa_n$ as the private key, where n is the length of the bit string. Then, the unconditionally secure deterministic secure quantum communication (DSQC) protocol[23] is used to write the private key in the blockchain to secretly share it. In this case, Jack secretly holds (ID, π) pair, where ID is Alice's identity code.

Step 2 Alice performs the $\rho_\pi^+(n)$ generation algorithm to obtain the public key $|PK\rangle = \otimes_{i=1}^n \rho_\pi^{i+}$. (One bit one key), as shown in Fig. 4a.

Step 3 Alice has a key pair $(|PK\rangle, \pi)$.

Signing phase. *Step 1* Alice performs a permutation π operation on TA and obtains the bit string $t : \pi(TA) = t$, where the $t = (t_1, t_2, \cdots, t_n), t_i \in \{0, 1\}$.

Step 2 Through the conversion algorithm, Alice encrypts t as a quantum sequence: $\rho = \otimes_{i=1}^n \rho_i$, where $\rho_i = \begin{cases} \rho_\pi^+(n), \text{if } t_i = 0 \\ \rho_\pi^-(n), \text{if } t_i = 1 \end{cases}$, as shown in Fig. 4b.

Step 3 Alice prepares r decoy particles ($r \gg 2n$), which are distributed randomly in $(|1\rangle, |0\rangle, |+\rangle, |-\rangle)$. She inserts r decoy particles randomly into $|\rho\rangle|PK\rangle$ and gets the sequence $|\rho'\rangle|PK'\rangle$ to check eavesdropping[8], as shown in Fig. 4c. She then sends $\{TA, ID, |\rho'\rangle, |PK'\rangle\}$ to Bob.

Step 4 After receiving $\{TA, ID, |\rho'\rangle, |PK'\rangle\}$, Alice exposes the location of the decoy particles. Bob checks the particles with the corresponding base. If there is no error, Bob takes the next step, and otherwise the signature generation phase is restarted.

Step 5 Bob performs an eavesdropping check, drops all the decoy particles and finally holds $\{TA, ID, |\rho\rangle, |PK\rangle\}$ as the quantum signature of Alice.

Verifying phase. *Step 1* Bob converts the public key $|PK\rangle$ to $|PK_m\rangle$ based on TA, where $|PK_m\rangle = \otimes_{i=1}^n \rho_{\pi,m}^i$ and $\rho_{\pi,m}^i = \begin{cases} \rho_\pi^+(n), \text{if } m_i = 0 \\ \rho_\pi^-(n), \text{if } m_i = 1 \end{cases}$, as shown in Fig. 4d.

Step 2:Bob prepares the r decoy particles ($r \gg 2n$), which are randomly in $(|1\rangle, |0\rangle, |+\rangle, |-\rangle)$. He randomly inserts the r decoy particles into $|\rho\rangle|PK_m\rangle$ to get the sequence $|\rho''\rangle|PK''\rangle$ to check for eavesdropping, as shown in Fig. 5a. He then sends $\{ID, |\rho''\rangle, |PK''\rangle\}$ to Jack.

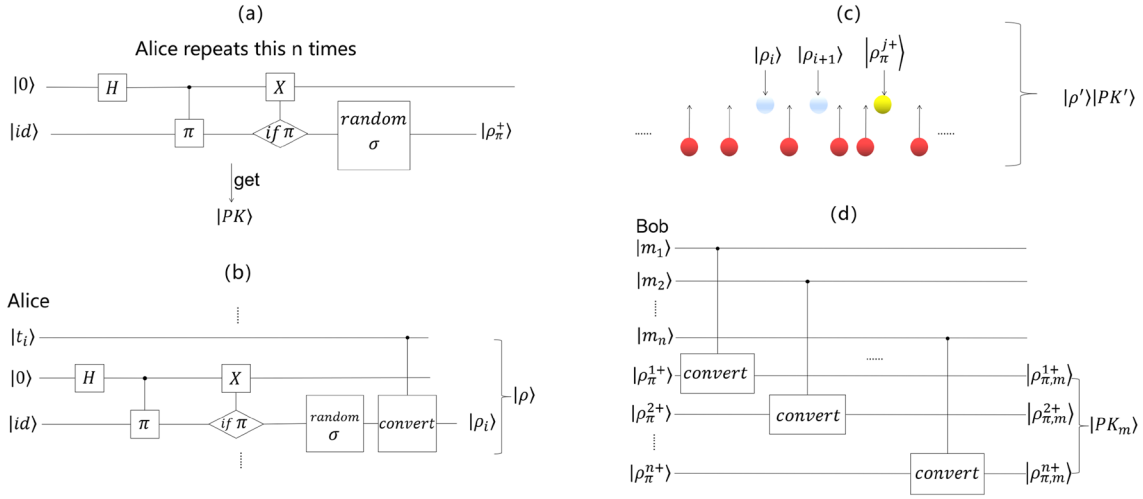

Figure 4. (**a**) Alice repeats the $\rho_{\pi}^{+}(n)$ generation algorithm n times with her private key to obtain the public key, where the H, "if π" and "random σ" operations are the same as the operations in Fig. 3. (**b**) Alice uses this quantum circuit to obtain encrypted sequence $|\rho\rangle$, where we perform a "C-convert" operation: if $|t_i\rangle = 1$, we perform the "convert" operation shown in Fig. 3c; if $|t_i\rangle = 0$, we do not perform any operation. (**c**) The red balls represent decoy particles, the white balls represent encrypted sequence $|\rho\rangle$, and the yellow balls represent public key $|PK\rangle$. Alice inserts decoy particles randomly into $|\rho\rangle|PK\rangle$ and obtains the sequence $|\rho'\rangle|PK'\rangle$ to check for eavesdropping. (**d**) Bob uses this quantum circuit to obtain $|PK_m\rangle$, where every "C-convert" operation is the same as the "C-convert" operation in (**b**).

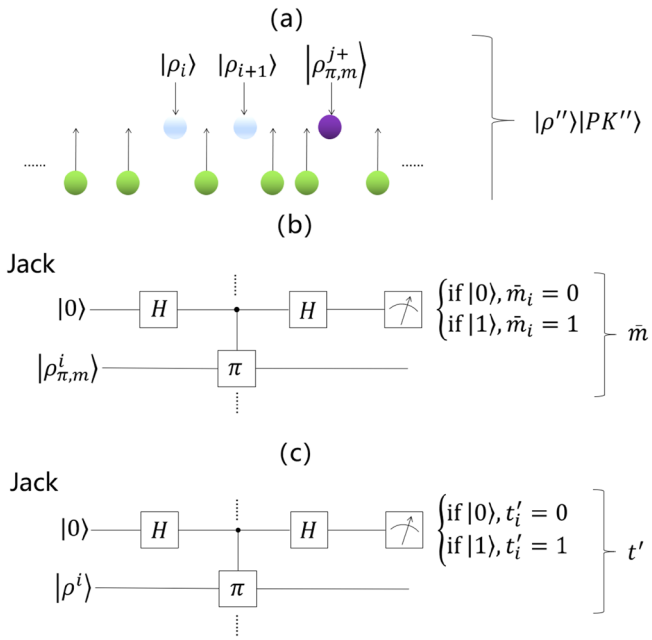

Figure 5. (**a**) The green balls represent decoy particles, the white balls represent encrypted sequence $|\rho\rangle$, and the purple balls represent public key $|PK\rangle$. Bob randomly inserts the decoy particles into $|\rho\rangle|PK_m\rangle$ to obtain the sequence $|\rho''\rangle|PK''\rangle$ to check for eavesdropping. (**b**) Jack uses this quantum circuit to obtain bit string $\bar{m}$. For every $\rho_{\pi,m}^{i}$, if $|0\rangle$ is read from the first register, then $\bar{m}_i = 0$; if $|1\rangle$ is read from the first register, then $\bar{m}_i = 1$. (**c**) Jack uses this quantum circuit to obtain a bit string t'. For every ρ_i, if $|0\rangle$ is read from the first register, then $\bar{t}_i = 0$; if $|1\rangle$ is read from the first register, then $\bar{t}_i = 1$.

Step 3 After Jack receives $\{ID, |\rho''\rangle, |PK''\rangle\}$, Bob exposes the location of the decoy particles. Jack checks the particles with the corresponding base. If there is no error, Jack takes the next step, and otherwise the signature generation phase is restarted.

Step 4 Jack discards all decoy particles and recovers $|\rho''\rangle|PK''\rangle$ to $|\rho\rangle|PK_m\rangle$.

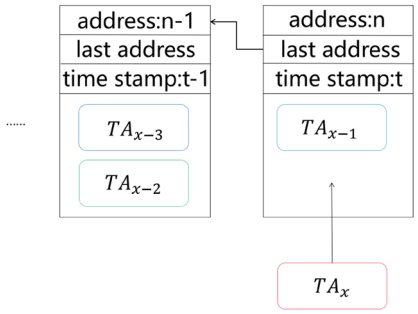

Figure 6. Transactions verified by 2/3 of witnesses can be packaged into a blockchain by the present witness, where the data structure of the blocks is the same as the data structure in Fig. 1, and TA_x is a valid transaction that needs to be packaged.

Step 5 Jack recovers the private key π according to the identity code ID, and obtains the bit string $\overline{m}$ by distinguishing $\rho^i_{\pi,m}$, where $\bar{m}_i = \begin{cases} 0, \rho^i_{\pi,m} = \rho^+_\pi(n) \\ 1, \rho^i_{\pi,m} = \rho^-_\pi(n) \end{cases}$, as shown in Fig. 5b. Then, permutation π is performed on $\overline{m}$, and $\bar{t} = \pi(\overline{m})$ is obtained.

Step 6 Jack distinguishes ρ_i, and obtains the bit string t', where $t' = \begin{cases} 0, if\ \rho_i = \rho^+_\pi(n) \\ 1, if\ \rho_i = \rho^-_\pi(n) \end{cases}$, as shown in Fig. 5 (c). If $\bar{t}_i = t'$, Jack claims validation and Bob accepts the signature.

Package the transaction into blockchain. In actual applications, the witness codes elected under DPoSB should be considered trusted signature verifiers. After more than 2/3 of the witness nodes accept the signature, the generated transaction information TA is valid and packed into the block generated by the current witness node, as shown in Fig. 6. However, when the verifying phase is completed, if less than 2/3 of the witness nodes accept the signature, TA is discarded by the current witness node.

Security analysis of the blockchain

Security model. Before reviewing the security of our blockchain, we would explain two security models used in information theory and cryptography[24].

Semi-honest adversary model Suppose there are some semi-honest adversaries in a system and they follow a protocol correctly but may keep some necessary information to infer additional information later.

Malicious adversary model Suppose there are some malicious adversaries in a system and they may not only keep necessary information to infer additional information, but also attempt to perform breaking-protocol malicious behaviours to get additional information.

In the block generation process, a semi-honest adversary can only keep public information of the block header and block body. Then he cannot infer any useful additional information, because there are no secrets in the public information. In the signing process, a semi-honest adversary can attempt to infer the private key of a signer (the only secret), which however cannot work as shown in "Security of private keys" Section. Therefore our blockchain can keep security in the semi-honest adversary model.

We will demonstrate the security in the malicious adversary model in the next three sections. Generally, when an algorithm or a protocol can keep security in the malicious adversary model it is safer.

Security of the generation of blocks. Consensus algorithms are used in the generation of blocks, and different consensus algorithms have distinct security levels. There are three main breaking-protocol attacks in this process, which belong to the malicious adversary model: 1. Double-spending attacks[1]. 2. Attacks that crack the hash value in a short time[14]. 3. Nodes that disturb the generation of blocks on purpose[13]. Then we will explain how our blockchain has robustness in the block generation process to these attacks in the malicious adversary model.

Attacker nodes can forge another blockchain secretly to forge information in blocks, which is defined as double-spending attacks. The success rate of this attack is higher when the computing force is larger. The success rate becomes 100% when the computing force of one node is larger than half of the total computing force of the blockchain system. However, this attack can be defended against in our blockchain algorithm because it is based on the computing force that is not needed in our algorithm.

An attack that cracks the hash value in a short time is a special attack based on a quantum computer. The quantum computer can use quadratic acceleration to crack the hash value through the Grover algorithm[25], which makes nodes that have quantum computers dominate the blockchain systems. However, this attack is still based on computing force, so it can be defended in our blockchain algorithm.

As shown in "Blocks created by DPoSB" Section, in blockchain systems, some nodes may intentionally disturb the generation of blocks. In the DPoSB algorithm, malicious behaviours can be recorded by blockchain systems, and these records have impacts on the nodes' scores during the elections. Because the chance that a

node is elected as a witness is smaller when it has more malicious behaviours, our blockchain algorithm can also defend against this attack.

Quantum information-theoretical security. In quantum asymmetric encryption, an encryption has quantum information-theoretical security if the quantum cyphertexts have computational indistinguishability[26].

We can claim that two quantum ensembles ρ_1 and ρ_2 are computationally indistinguishable, if for every probabilistic polynomial algorithm A, every positive polynomial $P(.)$ and sufficiently large positive integer n the following inequation can be satisfied [26]:

$$|P_r(A(\rho_1) = 1) - P_r(A(\rho_2) = 1)| < \frac{1}{P(n)},$$

where $P_r(.)$ represents the probability.

In our blockchain algorithm, the cyphertexts are $\rho_\pi^+(n)$ and $\rho_\pi^-(n)$. Then we define that $\rho_1 = \rho_\pi^+(n)^{\otimes P(n)}$, $\rho_2 = \rho_\pi^-(n)^{\otimes P(n)}$ and need to prove:

$$|P_r(A(\rho_\pi^+(n)^{\otimes P(n)}) = 1) - P_r(A(\rho_\pi^-(n)^{\otimes P(n)}) = 1)| < \frac{1}{P(n)}.$$

Assume that we have a probabilistic polynomial algorithm A_l, which makes:

$$|P_r(A_l(\rho_\pi^+(n)^{\otimes P(n)}) = 1) - P_r(A_l(\rho_\pi^-(n)^{\otimes P(n)}) = 1)| \geq \frac{1}{P(n)}.$$

It means we have an efficient algorithm to distinguish signature cyphertexts $\rho_\pi^+(n)$ from $\rho_\pi^-(n)$ efficiently, corresponding to solving the $\mathrm{QSCD}_{\mathrm{ff}}$ problem. However, according to the hardness of the $\mathrm{QSCD}_{\mathrm{ff}}$ problem as proved in ref.[22], the problem cannot be solved in polynomial time. Thus, it can be claimed that our blockchain has quantum information-theoretical security.

Security of the signing process. The malicious attacks which can be used in this process are eavesdropping, forging, repudiation and interception. Then we will explain how our blockchain can have robustness in the signing process to these attacks in the malicious adversary model.

Security of private keys. The security of private keys should be assured in two ways.

First, it has been proven that no quantum algorithm can crack the private keys of signers in polynomial time when there is no private key π [22] because one cannot distinguish signature cyphertexts $\rho_\pi^+(n)$ from $\rho_\pi^-(n)$ efficiently, as discussed in "Quantum information-theoretical security" Section.

Second, because private keys are selected from κ_n and $|\kappa_n| = \frac{n!}{\sqrt{2}^n}$, the attacker only has a chance of $\frac{\sqrt{2}^n}{n!}$ to obtain the private keys (note that the divergence of $n!$ is far stronger than $\sqrt{2}^n$). In this case, the success rate of brute attacks is sufficiently small, which means that the success rate of signatures randomly generated by attackers is negligible.

Security against eavesdropping. As mentioned above, we can use the BB84[13] protocol to defend against eavesdropping. Because of the particularity of quantum states, eavesdropping can result in the collapses of quantum states and destroy the decoy states. By the second checkout process in BB84, the verifier could determine if there is any eavesdropping through the measurement of decoy states. In addition, eavesdropping by cloning signatures is not possible because of the quantum no-cloning theorem[21].

Security against forging. There are two forging attack approaches. The first is forging signatures by using the transaction information of signers, and the second is forging the transaction information of signers.

In the first approach, a signer generates transaction information TA and public key $|PK\rangle$ and then uses the private key to generate signature $|\rho_1\rangle$. An attacker wants to forge a signature with TA and the signer's private key, which makes $|\rho_1\rangle \neq |\rho_2\rangle$. According to the signature algorithm mentioned above in section "Transaction signing and verification process", because of the uniqueness of the output of the $\rho_\pi^+(n)$ generating algorithm, we have $|\rho_1\rangle = |\rho_2\rangle$, and thus, the signatures cannot be forged.

In the second approach, a signer generates transaction information $TA1$ and public key $|PK\rangle$. An attacker wants to forge the signer's transaction information by turning it into $TA2 \neq TA1$ to make the signature of $TA2$ pass the verification process. According to the security of the private keys mentioned in section "Security of private keys", attackers have no way to generate a valid signature when they have no signers' private keys. Therefore, transaction information cannot be forged. In Conclusion, the forging methods mentioned above cannot be performed.

Security against repudiation. Repudiation is that attackers repudiate signatures to make signers fail in the signing process.

According to the signature algorithm mentioned above in section "Transaction signing and verification process", an attacker has no access to verify the signatures when they are not a witness; hence, an attacker cannot repudiate signatures. When an attacker is a witness, Jack can automatically pass through the signature if verification succeeds. In this way, an attacker still cannot repudiate the signatures because Jack is a trusted node and determines whether a signature can pass through the verification process.

Security against interception. Interception is that attackers forge information through intercepting information.

According to the signature algorithm mentioned in section "Transaction signing and verification process", messages, including $TA, ID, |\rho'\rangle, |PK'\rangle, |\rho''\rangle, |PK''\rangle$ and the location information of decoy particles, can be intercepted by an attacker.

In the signing phase,$TA, ID, |\rho'\rangle, |PK'\rangle$ is first intercepted. Then, to avoid the suspects of the signer, the attacker has to forge a new message,$TA1, ID, |\rho_1\rangle, |PK'\rangle$, to pass the verification process, which is a man-in-the-middle attack. According to the analysis mentioned above in section "Security against forging", even if the attacker passes through the decoy particle check process, the forging messages still cannot pass through the verification process because the attacker has no signer's private key.

In the verification phase, the attacker first intercepts $TA, ID, |\rho''\rangle, |PK''\rangle$. Then, to avoid the suspicion of the signer, the attacker has to forge a new message $TA2, ID, |\rho_2\rangle, |PK''\rangle$ to pass through the verification process, which is a man-in-the-middle attack too. In this way, the reason for a forging failure is the same as that for a signing phase failure.

Security issues from actual applications. In actual applications, there are several other security problems for businesses, organizations and operations. According to recent research progresses[27–30], some kinds of techniques, such as Process-Data-Infrastructure (PDI) model[27], can be incorporated into blockchain systems to figure out these problems and secure blockchain applications.

According to the PDI model[27], system security issues can be classified to three levels: process level, data level and infrastructure level. The blockchain security in the process level includes operation standards, smart contracts, implementation security and fraud detection. The data level is composed of consensus algorithms, encryption, authentication, key management and access control while the infrastructure level includes supernode server, terminal devices and network. In the above sections we have discussed the blockchain security issues in the data level, and our blockchain can be combined with the modern blockchain frame (such as the PDI model) to enhance the security of the blockchain system. In blockchain-secured smart manufacturing [28], a specific PDI model can be realized like the following architecture: in the infrastructure level, a blockchain platform (such as Ethereum, Hyperledger, and EOS) is selected to manage terminals and networks. The platform should provide distributed data structure, interaction mechanisms, and computing paradigms. Then in the data level, our blockchain algorithm can be used to generate blocks (by consensus algorithm) and sign the transactions (by quantum digital signature) safely. More complex computations are performed safely with privacy computing (such as secured multi-party computation[24], federated learning[31] and trusted execution environment[32]), which makes blockchain compute functions on private data with them unexposed. Then a computer language supported by the blockchain platform is used to write smart contracts in the process level. Programmable manufacturing devices can be deployed in necessary places, and relevant data are collected through internet of things (IoT)[33], which are transmitted to blockchain for next processing.

Comparison with other quantum blockchain signature methods

In actual applications of blockchain technology, security is of the most importance; thus, we would use the safest signature algorithm as much as possible. Then, we will demonstrate that the signature algorithm in our blockchain algorithm is safer than other quantum-resistant signature algorithms. We assume that the decoherences of quantum circuits with outside environments can be ignored.

Comparison with a signature algorithm based on nonorthogonal encoding. We introduce the basic ideas of nonorthogonal [20] encoding first. We define four quantum states: $|0\rangle, |1\rangle, |+\rangle, |-\rangle$, where $|0\rangle$ and $|1\rangle$ are eigenstates of Pauli Z and $|+\rangle$ and $|-\rangle$ eigenstates of Pauli X. The signer prepares a binary string $a = (a_1 a_2 ... a_n), a_i \in \{0, 1\}$, where n is large enough, and selects a trusted authenticator. Then, we define four nonorthogonal sets, $\{|0\rangle, |+\rangle\}, \{|+\rangle, |1\rangle\}, \{|1\rangle, |-\rangle\}, \{|-\rangle, |0\rangle\}$, and any two quantum states in each set are not orthogonal to each other. The signer, verifier and authenticator can perform the next procedures to complete this signature algorithm.

Step 1 The signer selects a random quantum state from the four quantum states and distributes the nonorthogonal sets containing this quantum state according to the corresponding bit in the code. For example, if $a_1 = 1$ and the signer selects $|0\rangle$, we distribute the set $\{|-\rangle, |0\rangle\}$ to the first quantum state; if $a_1 = 0$ and the signer selects $|0\rangle$, we distribute the set $\{|0\rangle, |+\rangle\}$ to the first quantum state. This process is repeated n times; then, the signer sends the quantum states $Q = (Q_1 Q_2 ... Q_n), Q_i \in \{|0\rangle, |1\rangle, |+\rangle, |-\rangle, \}$ and single bit information m to the verifier and the authenticator by quantum channels.

Step 2 The verifier and the authenticator choose an X or Z basis randomly for every quantum state Q_i and then take the measurements of these quantum states $\{Q_i\}$.

Step 3 The signer sends sets to the verifier and the authenticator by traditional channels.

Step 4 The verifier and the authenticator compare every result of quantum state Q_i with their sets. If one measurement result is orthogonal to one quantum state in the set, the conclusive bit a_i can be obtained. For example, if we receive set $\{|0\rangle, |+\rangle\}$ and the measurement result is $|-\rangle$, we can know that the signer sends 0; however, if the measurement result is not orthogonal to any quantum state in the set, the code the signer sent is inconclusive.

Step 5 The signer sends a bit string a to the verifier and authenticator. After receiving the bit string a, the verifier and the authenticator compare it with their conclusive bit string and compute the error rates $E(a')$ and $E(a'')$, respectively (we take inconclusive bits as right bits). If both $E(a')$ and $E(a'')$ are larger than threshold μ, the signature fails; otherwise the signature can succeed.

It can be demonstrated that this signature algorithm cannot defend against interception. An attacker can perform the next procedures to forge a signer's signature.

Step 1 The signer generates single bit information m, bit string a and quantum state Q and then sends them to the verifier and the authenticator.

Step 2 The attacker intercepts the messages $\{m, a, Q\}$; generates single bit information m', bit string a' and quantum state Q'; and then sends $\{m', a', Q'\}$ to the verifier and the authenticator.

Step 3 The signer sends sets $Q1$ of Q to the verifier and the authenticator.

Step 4 The attacker incepts messages $Q1$ and sends sets $Q2$ of Q' to the verifier and the authenticator.

Step 5 The verifier and the authenticator perform step 5 in the signature algorithm.

Step 6 Now, the attacker forges a perfect signature of the signer because it is simple to generate $\{m', a', Q'\}$ and $Q2$, so the verifier and the authenticator can pass the signature forged by the attacker with overwhelming probability. A forging attack can work in this way.

As mentioned in section "Security against interception", we have demonstrated that the signature algorithm in our blockchain algorithm can resist interception and thus is safer than the algorithm in this section.

Comparison with a signature algorithm based on quantum entanglement. Suppose there are three characters that take part in this algorithm [16]: the signer, the verifier and a trusted node blockchain. They perform the next procedures to complete this signature algorithm.

Step 1 The blockchain generates sufficient Bell states: $\{(A_1^1, A_1^2), (A_2^1, A_2^2), ..., (A_n^1, A_n^2)\}$, where (A_i^1, A_i^2) represents Bell state $|\psi\rangle_{A_i^1 A_i^2} = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$. Hence, we have two qubit strings $(A_1^1, A_2^1, ..., A_n^1)$ and $(A_1^2, A_2^2, ..., A_n^2)$. In the same way, the blockchain generates Bell states: $\{(B_1^1, B_1^2), (B_2^1, B_2^2), ..., (B_n^1, B_n^2)\}$, so we have two qubit strings $(B_1^1, B_2^1, ..., B_n^1)$ and $(B_1^2, B_2^2, ..., B_n^2)$.

Step 2 The blockchain randomly selects a sufficiently long substring A_1 from $(A_1^1, A_2^1, ..., A_n^1)$ and sends it to the signer as his private key; the blockchain randomly selects a sufficiently long substring A_2 from $(A_1^2, A_2^2, ..., A_n^2)$ and sends it to the verifier as the signer's private key; the blockchain randomly selects a sufficiently long substring B_1 from $(B_1^1, B_2^1, ..., B_n^1)$ and sends it to the verifier as the private key; the blockchain randomly selects a sufficiently long substring B_2 from $(B_1^2, B_2^2, ..., B_n^2)$ and sends it to the signer as the verifier's private key.

Step 3 The signer uses the hash function $(h = hash(m))$ on an x-length quantum coin $m = \{m_1, m_2, \ldots, m_x\}, m_i \in \{|0\rangle, |1\rangle\}$ to obtain a y-length hash sequence.

Step 4 The signer performs controlled-NOT(CNOT) on the first x qubits of B_2, the first y qubits of A_1, quantum coin m and hash sequence h:

$$CNOT_{B_{2i}m_i}|\psi\rangle_{B_{2i}B_{1i}}m_i = \frac{|00\rangle m_i + |11\rangle\overline{m}_i}{\sqrt{2}},$$

$$CNOT_{A_{1i}h_i}|\psi\rangle_{A_{2i}A_{1i}}h_i = \frac{|00\rangle h_i + |11\rangle\overline{h}_i}{\sqrt{2}},$$

where $\overline{m}_i = 1 - m_i$and$\overline{h}_i = 1 - h_i$. Then, the signer obtains quantum coin m' and hash sequence h' and sends m' and h' to the verifier.

Step 5 The verifier performs CNOT on the first x qubits of B_1, the first y qubits of A_2, quantum coin m and hash sequence h':

$$CNOT_{B_{1i}m_i}\frac{|00\rangle m_i + |11\rangle\overline{m}_i}{\sqrt{2}} = |\psi\rangle_{B_{2i}B_{1i}}m_i,$$

$$CNOT_{A_{2i}h_i}\frac{|00\rangle h_i + |11\rangle\overline{h}_i}{\sqrt{2}} = |\psi\rangle_{A_{2i}A_{1i}}h_i,$$

Then, the verifier obtains quantum coin m'' and hash sequence h'', computes $hash(m'')$ and judges if it is equal to h''. If $hash(m'') = h''$, the signature is accepted; otherwise, the signature is rejected.

It can also be demonstrated that this signature algorithm cannot defend against interception. An attacker can perform the next procedures to forge a signer's signature.

Step 1 As shown in the signature algorithm mentioned above, the blockchain generates substrings A_1, A_2, B_1, and B_2 and sends A_1 and B_2 to the signer and A_2 and B_1 to the verifier.

Step 2 The attacker intercepts A_2 and B_1 through a man-in-the-middle attack, imitates the blockchain to generate substrings C_1, C_2, D_1, and D_2, retains substrings C_1 and D_2, and then sends substrings C_2 and D_1 to the verifier.

Step 3 In this moment, the attacker's substrings entangle the signer's and the verifier's at the same time, so the attacker can forge any transaction messages and the signatures of the signer and the verifier.

Conclusion

We propose a quantum blockchain algorithm that generates blocks by DPoSB and signs the transaction information with a quantum one-way function based on the $QSCD_{ff}$ problem. By the stake vote and punishing the malicious behaviours of DPoSB and asymmetric quantum encryption, the fairness, efficiency and security of the blockchain system can be improved. Security in the semi-honest adversary model and the malicious adversary model can be realized in our blockchain based on quantum information-theoretical security. Furthermore, we demonstrate the security of our blockchain algorithm compared with other quantum blockchain algorithms. Our quantum blockchains provide a safe platform that could decrease the costs of various operations and transaction activities. We should mention that the trusted node used in our blockchain has a larger weight in the network

and therefore the necessity of the trusted node may weaken decentralization. Quantum signatures which do not require the trusted node could be developed in future researches to solve this problem. Moreover, quantum blockchains could be based on quantum privacy computing, which would further enhance the security of actual blockchain applications. In the near future, quantum blockchains will play an important role in social and financial areas that have increasing demands for transaction securities.

Data availability

The authors declare that the data that support the plots within this paper and other findings of this study are available from the corresponding author upon reasonable request.

Received: 1 October 2021; Accepted: 10 May 2022
Published online: 21 May 2022

References

1. Nakamoto, S. A peer-to-peer electronic cash system. *Decentralized Bus. Rev.* 21260 (2008).
2. King, S., Nadal, S. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper*. **19** (2012).
3. Larimer, D. Delegated proof-of-stake (dpos). *Bitshare Whitepaper.* **81**, 85 (2014).
4. Tan, C., Xiong, L. DPoSB: Delegated Proof of Stake with node's behavior and Borda Count. *ITOEC.* 1429–1434 (2020).
5. Lamport, L. The Byzantine generals problem. *JACM.* **30**, 668–676 (1983).
6. Rivest, R. L., Shamir, A. & Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM.* **21**, 120–126 (1978).
7. Miller, V. S. Use of elliptic curves in cryptography. *Conf. Theor. Appl. Cryptogr. Techn.*417–426 (1985).
8. Shor, P. W. Algorithms for quantum computation: discrete logarithms and factoring. *IEEE Proc. Annu. Symp. Found. Comput. Sci.* 124–134 (1994).
9. Arute, F. *et al.* Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
10. Pogorelov, I. *et al.* Compact Ion-trap quantum computing demonstrator. *PRX Quantum.* **2**, 020343 (2021).
11. Zhong, H. S. *et al.* Quantum computational advantage using photons. *Science* **370**, 1460–1463 (2020).
12. Arrazola, J. M. *et al.* Quantum circuits with many photons on a programmable nanophotonic chip. *Nature* **591**, 54–60 (2021).
13. Bennett, C. H., Brassard, G. Quantum cryptography: Public key distribution and coin tossing. Preprint at https://arxiv.org/abs/2003.06557 (2020).
14. Kiktenko, E. O. *et al.* Quantum-secured blockchain. *Quantum Sci. Technol.* **3**, 035004 (2018).
15. Rajan, D. & Visser, M. Quantum blockchain using entanglement in time. *Quantum Rep.* **1**, 3–11 (2019).
16. Gao, Y. L. *et al.* A novel quantum blockchain scheme base on quantum entanglement and DPoS. *Quantum Inf. Process.* **19**, 1–15 (2020).
17. Xin, X., Yang, Q. & Li, F. Quantum public-key signature scheme based on asymmetric quantum encryption with trapdoor information. *Quantum Inf. Process.* **19**, 1–15 (2020).
18. Chuang, I., Gottesman, D. Quantum digital signatures. Preprint at https://arxiv.org/abs/quant-ph/0105032 (2001).
19. Chen, F. L. *et al.* Public-key quantum digital signature scheme with one-time pad private-key. *Quantum Inf. Process.* **17**, 10 (2018).
20. Lu, Y. S. *et al.* Efficient quantum digital signatures without symmetrization step. *Opt. Express.* **29**, 10162–10171 (2021).
21. Nielsen, M. A. & Chuang, I. L. Quantum computation and quantum information. *Phys. Today.* **54**, 60 (2001).
22. Kawachi, A. *et al.* Computational indistinguishability between quantum states and its cryptographic application. *J. Cryptol.* **25**, 528–555 (2012).
23. Yan, L. *et al.* Semi-quantum protocol for deterministic secure quantum communication using Bell states. *Quantum Inf. Process.* **17**, 315 (2018).
24. Goldreich, O. Secure multi-party computation. *Manus. Prelim. Vers.* **78**, 110 (1998).
25. Grover, L. K. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.* **79**, 325 (1997).
26. Pan, J., Yang, L. Quantum public-key encryption with information theoretic security. Preprint at https://arxiv.org/abs/1006.0354 (2010).
27. Leng, J., Zhou, M., Zhao, J. L., Huang, Y. & Bian, Y. Blockchain security: A survey of techniques and research directions. *IEEE Trans. Serv. Comput.* https://doi.org/10.1109/TSC.2020.3038641 (2020).
28. Leng, J. *et al.* Blockchain-secured smart manufacturing in industry 4.0: A survey. *IEEE Trans. Syst. Man Cybern. Syst.* **51**, 237–252 (2020).
29. Berdik, D. *et al.* A survey on blockchain for information systems management and security. *Inf. Process. Manage.* **58**, 102397 (2021).
30. Muralidhara, S., Usha, B. A. Review of Blockchain Security and Privacy. *2021 5th IEEE ICCMC* 526–533 (2021).
31. Kim, H. *et al.* Blockchained on-device federated learning. *IEEE Commu. Lett.* **24**, 1279–1283 (2019).
32. Ayoade, G., Karande, V. & Khan, L. *et al*. Decentralized IoT data management using blockchain and trusted execution environment. *IEEE IRI* 15–22 (2018).
33. Hassan, M. U., Mubashir, H. R. & Chen, J. Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions. *Future Gener. Comput. Syst.* **97**, 512–529 (2019).

Acknowledgements

Work was supported by the Fundamental Research Funds for the Central Universities (Grant No. 14380146), National Natural Science Foundation of China (Grant No. 12074177, No.61521001, No.12074179 and No.11890704), the Key R&D Program of Guangdong Province (Grant No.2018B030326001) and the NKRDP of China (Grant No.2016YFA0301802).

Author contributions

L.D. and W.W wrote the main manuscript text. All authors reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to L.D.