International Journal of Software and Informatics, ISSN 1673-7288
http://www.ijsi.org, ijsi@iscas.ac.cn, +86-10-62661048
IJSI, 2025, 15(4): 495–519, doi: 10.21655/ijsi.1673-7288.00359

International Journal of Software and Informatics

Research Article

Post-quantum Blockchain System Based on Multi-parent Chain Auxiliary Proof-of-work Consensus Mechanism

Yutian Wang (王御天), Shurui Jia (贾舒瑞), Mingxi Chen (陈铭希), Yifan Dong (董怡帆), Yafang Yang (杨亚芳)

(School of Computer Science, Fudan University, Shanghai 200433, China)
Corresponding author: Yutian Wang, 23110240140@m.fudan.edu.cn

Abstract With the development of quantum computers, public blockchains relying on traditional elliptic curve digital signatures are expected to face disruptive security risks. A common solution involves the application of post-quantum digital signature algorithms within blockchain systems. For public blockchains utilizing the proof-of-work consensus mechanism, ensuring sufficient computing power is regarded as a critical foundation for security. Energy conservation and the maximization of computing power support have been identified as key research directions. Therefore, a post-quantum blockchain system featuring diversified computing power and autonomous post-quantum signature is proposed in this paper. The Dilithium signature scheme, recommended by the National Institute of Standards and Technology (NIST) as a preferred and general-purpose post-quantum signature standard, relies on the security of MLWE and MSIS problems in power-of-two cyclotomic rings. However, similar to the early adoption of the EC-DSA standard in Bitcoin without adherence to the NIST-specific elliptic curves, the rich algebraic structure of power-of-two cyclotomic rings poses greater risks and uncertainties regarding long-term security. To address this, a more conservative and secure approach, based on post-quantum lattice-based cryptography with fewer algebraic structures, is constructed. In this paper, a Dilithium variant, Dilithium-Prime, based on a large-Galois-group prime-degree prime-ideal field, is proposed as the signature algorithm for the post-quantum blockchain system to ensure high-confidence transaction signing with post-quantum security. To maximize the computing power support for the post-quantum public blockchain and address the current issue of declining mining pool and miner income, a multi-parent chain auxiliary proof-of-work consensus mechanism is introduced. This mechanism enables the request of computing power from all miners using Sha256 and Scrypt hash calculations to assist in consensus without increasing the workload for existing miners and mining pools. As a result, the source of computing power for the post-quantum blockchain is expanded, and the utilization rate of existing mining pools and miners is improved. In addition, a block and transaction structure, along with a difficulty adjustment algorithm tailored for this multi-parent chain auxiliary proof-of-work consensus mechanism, is proposed. This system stabilizes the block production ratio

This is the English version of the Chinese article “基于多父链辅助工作量证明共识机制的后量子区块链系统. 软件学报, 2025, 36(10): 4507–4524. doi: 10.13328/j.cnki.jos.007391”
Funding items: National Key Research and Development Program (2022YFB2701601); Collaborative Innovation Fund of Shanghai (XTCX-KJ-2023-54); Blockchain Key Technology Research and Development Special Fund of Shanghai Science and Technology Commission (23511100300)
Received 2024-06-30; Revised 2024-09-05; Accepted 2024-12-30; IJSI published online 2025-12-30

and production time across different levels of computing power and effectively responds to extreme cases, such as sudden surges or reductions in computing power, ensuring the system's robustness.

Keywords blockchain; post-quantum cryptography; consensus mechanism; auxiliary proof-of-work

Citation Wang YT, Jia SR, Chen MX, Dong YF, Yang YF. Post-quantum blockchain system based on multi-parent chain auxiliary proof-of-work consensus mechanism, *International Journal of Software and Informatics*, 2025, 15(4): 495–519. http://www.ijsi.org/1673-7288/359.htm

Blockchain technology was first proposed by Nakamoto in 2008[1]. Based on digital signature algorithms and consensus mechanisms, it possesses properties such as decentralization and immutability and is widely used in digital asset management, cross-border payments, and data sharing. Among these, Bitcoin, as the most widely recognized public blockchain, boasts a large number of users and a range of application scenarios. Concurrently, other public blockchains like Litecoin have emerged, adopting the same architecture and consensus algorithm as Bitcoin.

Public blockchains like Bitcoin employ the elliptic curve digital signature algorithm (ECDSA). During transactions, users sign with their private keys and submit both the data and signature to the chain, effectively ensuring the security and correctness of blockchain data. However, with the advent of quantum computers, the security of traditional cryptography like ECDSA cannot be guaranteed in the quantum era, which will severely impact the security of blockchain systems.

Furthermore, blockchain systems like Bitcoin use the proof-of-work (PoW) consensus mechanism[1]. The core concept of this algorithm is to compute a hash value that meets a certain difficulty threshold; the node that succeeds in this computation earns the right to produce a block. While the PoW consensus mechanism offers advantages like a high degree of decentralization, it also has drawbacks such as low utilization rates and high energy consumption. Following Bitcoin's fourth halving in April 2024, the profitability of most existing mining pools and miners has significantly decreased, creating an urgent need to diversify their revenue streams without additional investment in computational power. Moreover, after the emergence of quantum computers, more computational power will be required to enhance blockchain security.

Moreover, the consensus mechanism requires a difficulty adjustment algorithm (DAA) to maintain the average block production time as close as possible to a target value, thereby ensuring the stability of the blockchain system. However, due to its long adjustment period, the DAA struggles to respond promptly to sudden changes in the network's total hashrate. Sudden surges or drops in computational power can adversely affect system stability.

In response to the aforementioned background and problems, we design a blockchain system based on PoW consensus and UTXO script-based public blockchains. This system incorporates post-quantum digital signature algorithms and supports multiple computational power sources for consensus, with the following three main contributions.

(1) We design and implement the multi-parent chain auxiliary proof-of-work (Mul-AuxPoW) consensus mechanism. This mechanism allows other public blockchains that use Sha256 and Scrypt hash algorithms in their PoW to act as parent chains, providing computational power support. The advantage is the reuse of energy consumed by parent chains for consensus; the child chain itself requires no additional computational power or energy consumption. This also addresses the predicament of declining profitability for existing mining pools and miners, enabling them to generate additional revenue without extra investment in computational power and energy.

(2) Based on the Mul-AuxPoW consensus mechanism, we design a DAA. This algorithm ensures that the block production ratio from parent chains of different computational power sources remains within a certain proportion to incentivize participation from smaller computational power sources. By calculating the block boundary value, the algorithm adjusts the difficulty more flexibly, effectively coping with sudden computational power fluctuations.

(3) We apply an efficient digital signature scheme based on a large-Galois-group prime-degree prime-ideal field in the blockchain: the Dilithium-Prime algorithm[2]. Compared to the Dilithium post-quantum signature standard based on power-of-two cyclotomic rings, it has a simpler algebraic structure and a higher confidence level. In the algorithm's application, we expand the blockchain's transaction script to adapt to the post-quantum digital signature algorithm.

Section 1 of this paper introduces related work on the PoW consensus mechanism and the application of post-quantum signatures in blockchain systems. In Section 2, we describe the system architecture and relevant implementation details. A theoretical analysis of the security and correctness of the Mul-AuxPoW consensus mechanism and DAA are provided, along with an efficiency analysis, in Section 3. In Section 4, we present simulation experiments and evaluation of the system. Section 5 presents the conclusion of this paper.

1 Related Work

1.1 Application of post-quantum signature algorithms in blockchain

In 1994, Shor's algorithm was proposed[3], which can solve the discrete logarithm problem and the large integer factorization problem in polynomial time. This implies that for quantum computers, current cryptographic algorithms will face security challenges. In 2016, the National Institute of Standards and Technology (NIST) initiated a post-quantum cryptography standardization project[4]. The submitted cryptographic schemes included those based on lattices, codes, hashes, and others. Among them, lattice-based post-quantum cryptographic algorithms include digital signature schemes such as CRYSTALS-Dilithium[5] and Falcon[6].

Compared to other types, lattice-based post-quantum digital signatures, such as CRYSTALS-Dilithium, generally offer faster speeds but incur increased space overhead. Dong *et al*.[2] improved the Dilithium scheme, proposing a digital signature scheme based on a large-Galois-group prime-degree prime-ideal field, denoted as Dilithium-Prime. Compared to Dilithium, Dilithium-Prime features smaller memory overhead and stronger signing efficiency, and it offers improved security.

There have also been explorations into applying post-quantum signature algorithms in blockchain. For lattice-based post-quantum signature algorithms, Torres *et al*.[7] designed a lattice-based ring signature and applied it to Monero; Arielcoin employed the CRYSTALS-Dilithium signature algorithm in a PoW-based public blockchain system[8], primarily using the Dilithium3 parameters; Hcash adopted a distributed structure based on a directed acyclic graph (DAG)[9], utilizing a hybrid PoW/PoS consensus and a public key encryption scheme based on ring-LWE.

Regarding other types of post-quantum signature algorithms, there have also been attempts to apply them in consensus mechanisms. For example, the ABC chain applied the Rainbow signature algorithm[10] and used a PoW consensus mechanism, but it is based on the NP-hard problem of multivariate polynomials; QRL used a hash-based forward-secure signature scheme[11], employing WOTS+ as the main building block; similarly, Mochimo used XMSS+[12] and proposed and verified a WOTS+ variant for one-time signatures.

1.2 PoW consensus mechanism

The consensus mechanism in a blockchain determines how participating nodes reach agreement on block data[13], specifically governing the process of electing block producers, generating blocks, and synchronizing them across the network. In public blockchain systems like Bitcoin and Litecoin, the PoW consensus mechanism is employed. PoW utilizes one-way hash function computations; a node gains the block production right when it computes a hash value meeting a specified difficulty target. Here, the *nonce* is the value iterated through to satisfy the formula; the *hashBlockHeader* is the hash of the block header, and the *Target* is the value inversely proportional to the difficulty.

$$Hash(nonce + hashBlockHeader) < Target$$

With technological evolution, the PoW consensus mechanism has been widely adopted due to its advantages, including strong decentralization and high security. Blockchains like Bitcoin and BCH use the Sha256 hash function in their consensus. Their combined computational power has grown from approximately 10 TH/s in 2012 to around 500 EH/s today, dominating the current global computational power landscape. Litecoin and Dogecoin use the Scrypt hash function in their consensus, with their aggregate computational power currently reaching about 1 PH/s. Furthermore, to enhance computational power utilization efficiency, Litecoin implemented a hard fork in 2014 to support the AuxPoW consensus mechanism.

The AuxPoW consensus mechanism[14] improves upon the basic PoW framework, enabling simultaneous consensus across two blockchains. The blockchain requiring auxiliary consensus is called the child chain, while the blockchain assisting the child chain's consensus is termed the parent chain. However, current AuxPoW implementations only support identical hash functions between parent and child chains, as demonstrated by Litecoin and Dogecoin, which both utilize Scrypt.

1.3 DAA

In the PoW consensus mechanism, the network's computational power fluctuates constantly. To maintain a stable average block production time, the difficulty must be periodically adjusted. Conventionally, a larger *Target* value implies easier computation and lower difficulty, while a smaller *Target* indicates higher difficulty. Bitcoin employs the following formula for difficulty adjustment[15]. Here, *actualTimespan* represents the actual time elapsed to produce a certain number of blocks, and *PoWTargetTimespan* denotes the desired target time to produce that same number of blocks.

$$newTarget = prevTarget \times \frac{actualTimespan}{powTargetTimespan}$$

By simultaneously limiting the adjustment multiplier per cycle to a factor of 4, the new difficulty value must satisfy the following inequality:

$$0.25 \times prevTarget \leq newTarget \leq 4 \times prevTarget$$

DAA often exhibits inherent lag, as it requires a certain time period to enact adjustments, while changes in computational power can occur rapidly. A common exploit of this characteristic is the pool-hopping attack[16]. In this scenario, miners participating in consensus selectively mine the blockchain that offers higher profits at a given moment. When the profitability of that chain decreases, they switch their consensus power to another chain, leading to instability in the block-producing computational power. There are two fundamental attack modes: the first is to increase computational power within a certain period, causing the *actualTimespan* to decrease; the difficulty formula will then increase the difficulty. Since the increased difficulty may lead to insufficient profitability, a pool-hopping attack is initiated. In the subsequent period, the computational power decreases, while the increased difficulty will cause the time of this period to increase, affecting the blockchain's block production efficiency. The second is to reduce computational power within a certain period, causing the *actualTimespan* to increase; the difficulty formula will then decrease the difficulty. After the difficulty is reduced, a pool-hopping attack is initiated; in the subsequent period, the computational power increases, allowing the attacker to obtain additional profits, and the reduced difficulty will cause this time to decrease, compromising the blockchain's stability.

2 Blockchain System Design

This section elaborates on the system from four aspects: system architecture, the Mul-AuxPoW consensus mechanism, the DAA, and the application of the post-quantum signature algorithm. The architecture subsection illustrates the block generation, consensus, and difficulty adjustment process. The Mul-AuxPoW subsection details the consensus workflow between the mining pool, parent chains, and the child chain. The DAA part explains the algorithm's workflow and application. Finally, the post-quantum signature part describes modifications for key generation, transaction signing, and block expansion.

2.1 System architecture

The architecture of the post-quantum signature blockchain based on Mul-AuxPoW is largely similar to common blockchain systems. As shown in Figure 1, it consists of five layers from bottom to top: storage layer, network layer, consensus layer, protocol layer, and application layer.

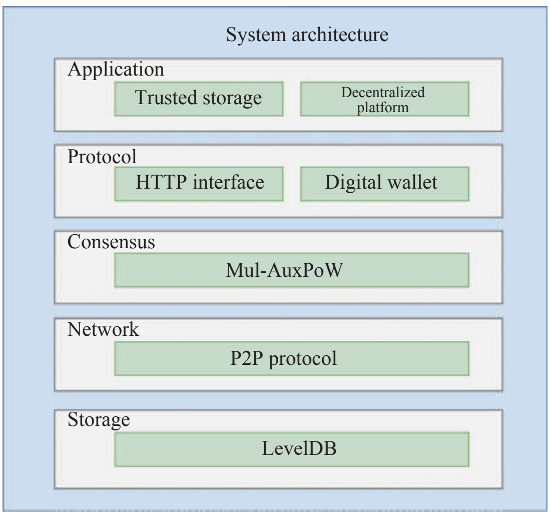

Figure 1 System architecture

- Storage layer: this layer redefines the block data structure. It utilizes the LevelDB database at the underlying level, storing blocks secured by both the Sha256 and Scrypt hash function consensus mechanisms, as well as transactions employing the Dilithium-Prime post-quantum digital signature algorithm.
- Network layer: this layer employs the peer-to-peer (P2P) protocol. Nodes communicate directly in a point-to-point manner, eliminating the need for a centralized server. The

P2P protocol provides the mechanisms for node connectivity and message routing.

- Consensus layer: this layer implements the Mul-AuxPoW consensus mechanism. For this mechanism, it leverages the computational power of two major categories of PoW blockchain systems as parent chains. Different difficulty targets are set for these two parent chain types. Based on principles we establish for efficiency, rationality, and fairness, a novel DAA automatically adjusts the difficulty for each corresponding parent chain.
- Protocol layer: this layer provides the HTTP interface for receiving transactions and returning results. It also includes modules such as the digital wallet, that are essential for using and upgrading the blockchain system.
- Application layer: this layer typically refers to the applications built on top of the blockchain system, encompassing various functionalities such as trusted storage.

2.2 Mul-AuxPoW consensus mechanism

The consensus mechanism primarily adheres to the "longest chain rule", similar to other blockchains using PoW. Its core principle is that each node always selects and attempts to extend the blockchain representing the greatest cumulative PoW. To minimize forks, a target block production time is set, and the block size is increased. If a fork occurs, the chain with the greatest block height is selected according to the longest chain rule.

The Mul-AuxPoW algorithm enables all blockchains using either the Scrypt or Sha256 hash algorithm in their consensus mechanism to act as parent chains, providing computational power to assist the child chain. As the majority of computational power in current consensus mechanisms is concentrated within mining pools, a model involving multiple parent chains, mining pools, and the child chain is designed to accommodate this characteristic. The specific workflow is illustrated in Figure 2.

The algorithm comprises four stages. Stage 1 is requesting a multi-parent chain auxiliary consensus. Parent chains using Scrypt and Sha256 hash algorithms in their consensus mechanism provide their types to the mining pool. The mining pool requests block templates containing different difficulties from the child chain according to the parent chain's type. The mining pool writes the hash value and difficulty from the child chain block template corresponding to the parent chain type into the parent chain's CoinBase transaction (the block's generation transaction, which generally does not contain blockchain transaction information), constructing the parent chain's block template for consensus.

Stage 2 is the auxiliary consensus stage. After the parent chain constructs the block template, it now has two difficulty values. The first difficulty value is the one in the parent chain block header used for the parent chain's block production. The second difficulty value is the one in the CoinBase transaction used for the child chain's auxiliary consensus. Parent chain consensus nodes will check whether both difficulty values are satisfied during the hash computation for consensus. If the child chain's difficulty value is met, the work is submitted to the child chain; if the parent chain's own difficulty requirement is met, the parent chain node obtains the block production right and proceeds to finalize the block template.

Stage 3 is the consensus submission stage. The mining pool provides the parent chain block or block template that meets the child chain's difficulty requirement to the child chain. This submission includes the following information: parent chain block header, Merkle branch of the CoinBase transaction, and CoinBase transaction field. The child chain modifies the data structure, saving the information to the *CAuxPoW* field pointed to by the *hashAuxPoW* field in the child chain block header.

Stage 4 is the work verification stage. After the child chain saves the three parts of information from the parent chain, it stores and verifies the information: (1) verify whether the child chain block header hash in the *CoinBaseTx* of the *CAuxPoW* field is the same as the block header hash and difficulty value submitted by the child chain to the mining pool. This step is used to verify whether the information submitted by the child chain is consistent with the CoinBase in the parent chain; (2) verify whether the root hash of the parent chain block transaction's Merkle tree, calculated from the *CoinBaseTx* of the *CAuxPoW* field and the corresponding *merkleBranch* is the same as the root hash value in the *hashBlockHeader*. This step is used to verify whether the CoinBase transaction containing correct information is in the parent chain that produced the block; (3) determine which type of parent chain the work comes from based on the *nVersion* field in the *hashBlockHeader* of the *CAuxPoW* field, select the corresponding submitted difficulty value to perform the hash verification of the corresponding parent chain on the *hashBlockHeader*, and determine whether it meets the difficulty requirement.

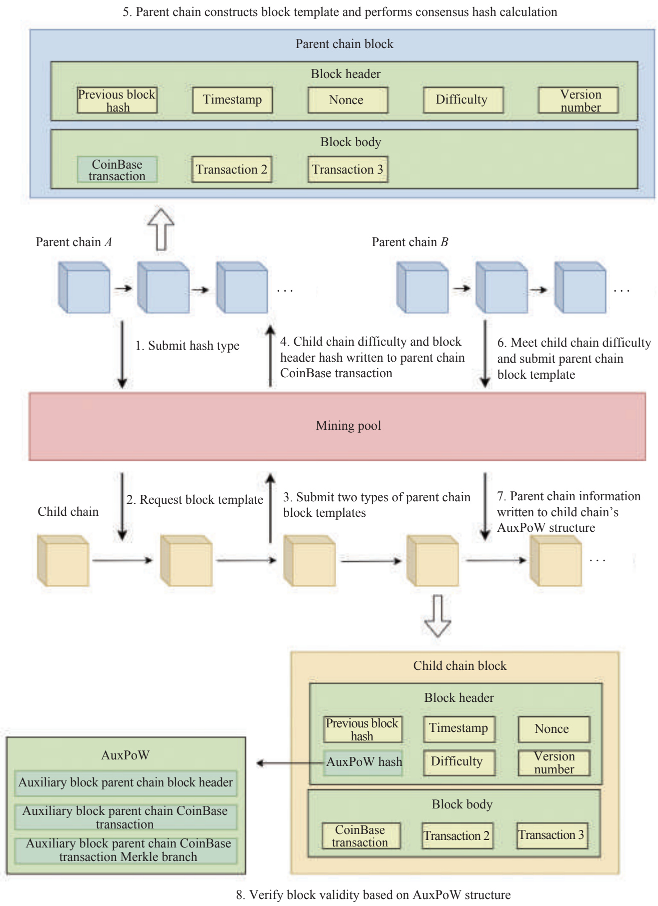

Figure 2 Mul-AuxPoW consensus mechanism flowchart

2.3 DAA

Currently, computational power is mainly concentrated in blockchains using the Sha256 hash function such as Bitcoin and BCH and blockchains using the Scrypt hash function such as Litecoin and DOGE. However, the computational power gap between the two types of chains remains relatively large and is concentrated in some mature blockchains. To attract blockchains with smaller computational power to join while not reducing the enthusiasm of chains with large computational power, and based on actual needs, the child chain must meet the following three requirements.

(1) Similar to other blockchains using the PoW consensus mechanism, the DAA needs to stabilize the number of blocks produced by the child chain around a constant φ within ET time (where ET and φ are constants).
(2) To protect parent chain B with relatively smaller computational power, the block production ratio between parent chain A (using the Sha256 hash function type) with larger computational power and parent chain B (using the Scrypt hash function type) should be maintained around a constant α.
(3) The child chain can resist the impact of sudden changes in computational power on the stability of block production time.

Based on these three requirements, a difficulty adjustment consensus mechanism is designed. We let the two types of parent chains calculate their difficulty values H_{A_n} and H_{B_n} for the n-th difficulty adjustment cycle based on their respective number of blocks produced x_n and y_n and the time within the cycle. Based on Bitcoin's DAA, two coefficient factors are added, representing the impact of computational power changes on difficulty and the block production ratio of the two parent chain types.

First, based on the number of blocks $x_{n-\omega+1}\ldots x_{n-1}x_n$ and $y_{n-\omega+1}\ldots y_{n-1}y_n$ produced by parent chains A and B, respectively, in the recent ω cycles, we calculate their respective weighted averages of block production change x_A and y_B. To reflect the change in computational power of the two chain types, we use the ratio of the difference in blocks produced between cycle $n-1$ and cycle n for chain L and the weighted average change in blocks produced over the previous ω cycles for chain L as the exponent and δ_T as the base, denoted as T_{A_n} and T_{B_n}, yielding:

$$T_{A_n} = \delta_T^{\frac{x_{n-1}-x_n}{x_A}}$$
$$T_{B_n} = \delta_T^{\frac{y_{n-1}-y_n}{y_B}}$$

Among these, more recent blocks better reflect changes in computational power, and their cycle difference carries greater weight. If $x_A = 0$, then $x_A = x_n - x_{n-1}$. Simultaneously, when x_A and $|x_n - x_{n-1}|$ are both relatively small, i.e., when the block production of chain A is stable over several cycles, the algorithm ensures T_{A_n} tends towards 1, reducing the influence of this parameter. The calculations for y_B and T_{B_n} are analogous.

Next, the number of blocks x_n and y_n, assisted by the two types of parent chains for the child chain within the target φ blocks, is counted. To maintain a stable block production ratio, two control methods are employed. For parent chain A, the first method involves comparing the ratio $\dfrac{x_n}{y_n}$ to the target block production ratio α, i.e., calculating $\alpha/(x_n/y_n)$. To prevent excessive fluctuation of this value from affecting stability, a boundary $[\mu, \xi]$ is set, specifically calculating $\max\left[\min\left(\alpha\cdot\dfrac{y_n}{x_n},\mu\right),\xi\right]$. For parent chain B, the analogous calculation is $\max\left[\min\left(\dfrac{x_n}{\alpha\cdot y_n},\mu\right),\xi\right]$.

The second method calculates the ratio of x_n to the target value φ and its relationship to the target ratio $\dfrac{\alpha}{\alpha+1}$, i.e., $(\alpha/\alpha+1)/(x_n/\varphi)$. To prevent excessive fluctuation of this value from affecting stability, a boundary $[\mu, \xi]$ is set, specifically $\max\left[\min\left(\dfrac{\varphi\cdot\alpha}{x_n(\alpha+1)},\mu\right),\xi\right]$.

However, it can be observed that $\frac{\varphi \cdot \alpha}{x_n(\alpha+1)}$ will not fall below $\frac{\alpha}{(\alpha+1)}$, and thus this can be simplified to $\min\left(\frac{\varphi \cdot \alpha}{x_n(\alpha+1)}, \mu\right)$. For parent chain B, the analogous calculation is $\max\left[\min\left(\frac{\varphi}{y_n(\alpha+1)}, \mu\right), \xi\right] \times \gamma$. In the algorithm, these two methods are combined, with assigned weighting coefficients β and γ, and $\beta + \gamma = 1$.

Finally, analogous to Bitcoin's difficulty adjustment method, the ratio of the actual block production time AT for φ blocks to the target block production time ET is calculated, i.e., $\frac{AT}{ET}$. At this stage, the result needs to be evaluated. When $\frac{AT}{ET} > 1.05$ or $\frac{AT}{ET} < 0.95$, it is considered that the block production time fluctuation is significant. In this case, stabilizing the block production time is more critical, and it is necessary to compare the product of the new factors to 1, ensuring the direction of difficulty adjustment does not change. When $0.95 < \frac{AT}{ET} < 1.05$, the block production time is considered relatively stable, and controlling the result of the new factors is unnecessary.

In summary, the specific difficulty adjustment formulas are as follows:

If $\frac{AT}{ET} > 1.05$,

$$
\begin{aligned}
H_{A_{n+1}} = H_{A_n} \cdot \frac{AT}{ET} &\\
\cdot \max\left\{\left\{\max\left[\min\left(\alpha \cdot \frac{y_n}{x_n}, \mu\right), \xi\right] \times \beta + \min\left(\frac{\varphi \times \alpha}{x_n(\alpha+1)}, \mu\right) \times \gamma\right\} \times T_{A_n}, 1\right\}&\\
H_{B_{n+1}} = H_{B_n} \cdot \frac{AT}{ET} &\\
\cdot \max\left\{\left\{\max\left[\min\left(\frac{x_n}{\alpha \cdot y_n}, \mu\right), \xi\right] \times \beta + \max\left[\min\left(\frac{\varphi}{y_n(\alpha+1)}, \mu\right), \xi\right] \times \gamma\right\}\right.&\\
\left. \times T_{B_n}, 1\right\}&
\end{aligned}
$$

If $0.95 < \frac{AT}{ET} < 1.05$,

$$
\begin{aligned}
H_{A_{n+1}} = H_{A_n} \cdot \frac{AT}{ET} &\\
\cdot \left\{\max\left[\min\left(\alpha \cdot \frac{y_n}{x_n}, \mu\right), \xi\right] \times \beta + \min\left(\frac{\varphi \times \alpha}{x_n(\alpha+1)}, \mu\right) \times \gamma\right\} \times T_{A_n}&\\
H_{B_{n+1}} = H_{B_n} \cdot \frac{AT}{ET} &\\
\cdot \left\{\max\left[\min\left(\frac{x_n}{\alpha \cdot y_n}, \mu\right), \xi\right] \times \beta + \max\left[\min\left(\frac{\varphi}{y_n(\alpha+1)}, \mu\right), \xi\right] \times \gamma\right\} \times T_{B_n}&
\end{aligned}
$$

If $\frac{AT}{ET} < 0.95$,

$$
\begin{aligned}
H_{A_{n+1}} = H_{A_n} \cdot \frac{AT}{ET} &\\
\cdot \min\left\{\left\{\max\left[\min\left(\alpha \cdot \frac{y_n}{x_n}, \mu\right), \xi\right] \times \beta + \min\left(\frac{\varphi \times \alpha}{x_n(\alpha+1)}, \mu\right) \times \gamma\right\} \times T_{A_n}, 1\right\}&
\end{aligned}
$$

$$
\begin{aligned}
H_{B_{n+1}} = H_{B_n} \cdot \frac{AT}{ET} \\
&\cdot \min\left\{\left\{\max\left[\min\left(\frac{x_n}{\alpha \cdot y_n}, \mu\right), \xi\right] \times \beta + \max\left[\min\left(\frac{\varphi}{y_n(\alpha+1)}, \mu\right), \xi\right] \times \gamma\right\}\right. \\
&\left. \times T_{B_n}, 1\right\}
\end{aligned}
$$

Based on the aforementioned algorithm, it is also necessary to ensure that the ratio between $H_{A_{n+1}}$ and H_{A_n}, as well as between $H_{B_{n+1}}$ and H_{B_n} falls within the range [1/5, 5], thereby adding upper and lower bounds to the difficulty adjustment. To defend against sudden computational power surge/drop attacks and combined attacks targeting type A and B chains, specific exceptional scenarios are defined based on the recent ω cycles and the count of cycles c_{less} and c_{more} where the number of blocks produced by each chain falls below $B_{\text{low}_{\text{limit}}}$ and $A_{\text{low}_{\text{limit}}}$.

(1) If $y_n \leq B_{\text{low}_{\text{limit}}}$, this indicates that the number of blocks assisted by the parent chain B with smaller computational power is relatively low. The reason could be a significant decrease in the computational power of parent chain B or a significant increase in the computational power of parent chain A. In this case, if $\frac{AT}{ET} \geq 1$, it can be concluded that the overall computational power is decreasing, implying that parent chain B is reducing its computational power. If this situation occurs consecutively for multiple cycles, it can be interpreted as parent chain B executing a computational power attack. In this scenario, the difficulty adjustment for parent chain B is set to be smoother, dampening the stronger influence of the new factors, thereby reducing the potential gains from launching a computational power attack under these specific conditions. Besides this type of attack, to prevent the new factors from having an excessively strong impact on difficulty adjustment and to reduce the magnitude of the adjustment, $H_{B{n+1}}$ needs to be adjusted using the following formula:

If $\frac{AT}{ET} \geq 1$ and $c_{\text{less}_B} \geq \omega - 1$, then: $H_{B_{n+1}} = H_{B_n} \cdot \log_\delta(\delta + \frac{AT}{ET})$

$$
\begin{aligned}
\text{Otherwise: } H_{B_{n+1}} = H_{B_n} \cdot \log_\delta\left(\delta + \frac{AT}{ET} \cdot \left\{\max\left[\min\left(\frac{x_n}{\alpha \cdot y_n}, \nu\right), \sigma\right] \times \beta\right.\right. \\
\left.\left. + \max\left[\min\left(\frac{\varphi}{y_n(\alpha+1)}, \mu\right), \xi\right] \times \gamma\right\} \times T_{B_n}\right)
\end{aligned}
$$

(2) If $y_n \geq A_{\text{low}_{\text{limit}}}$, similarly to the above, it indicates that the computational power of parent chain A has significantly decreased, or the computational power of parent chain B has significantly increased. To defend against computational power attacks, $H_{A{n+1}}$ is adjusted using the following formula:

If $\frac{AT}{ET} \geq 1$ and $c_{\text{more}B} \geq \omega - 1$, then $H_{A_{n+1}} = H_{A_n} \cdot \log_\delta(\delta + \frac{AT}{ET})$

$$
\begin{aligned}
\text{Otherwise, } H_{A_{n+1}} = H_{A_n} \cdot \log_\delta\left(\delta + \frac{AT}{ET} \cdot \left\{\max\left[\min\left(\alpha \times \frac{y_n}{x_n}, \mu\right), \xi\right] \times \beta\right.\right. \\
\left.\left. + \min\left(\frac{\varphi \times \alpha}{x_n(\alpha+1)}, \mu\right) \times \gamma\right\} \times T_{A_n}\right)
\end{aligned}
$$

Furthermore, two parameter sets are provided, as shown in Table 1. The first parameter set has a cycle length of 720 blocks with a target block production time of 2 minutes per block. This configuration features a relatively short cycle length, enabling more agile difficulty adjustment.

It is recommended for blockchains requiring rapid difficulty adjustment and for testing purposes. The second parameter set has a cycle length of 2,016 blocks with a target block production time of 5 minutes per block, featuring a relatively long cycle length. This aligns with parameters commonly used in existing public blockchains employing the PoW consensus mechanism and is recommended for more mature public blockchain applications.

Table 1 DAA parameters

Parameters	Target blocks per cycle φ	Proportion of blocks generated by parent chain α	Factor's upper bound μ	Factor's lower bound ξ	Weighting coefficient β	Weighting coefficient γ
1	720	10	3	0.33	0.4	0.6
2	2,016	10	3	0.33	0.5	0.5
Parameters	**Target time $ET(\text{min})$**	**Block record cycle ω**	**Base δ**	**Base δ_A**	**Block count boundary of parent chain $A_{\text{low}_\text{limit}}$**	**Block count boundary of parent chain $B_{\text{low}_\text{limit}}$**
1	1,440	4	10	1.2	320	5
2	10,080	3	10	1.2	1,516	6

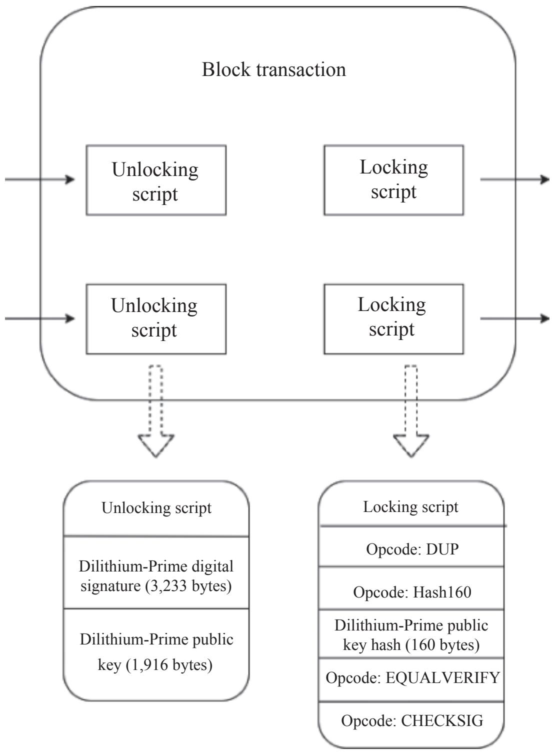

Figure 3 Block transaction script structure

2.4 Application of post-quantum digital signature algorithm based on a large-Galois-group prime-degree prime-ideal field

In blockchain systems utilizing the PoW consensus mechanism, transactions are constructed using scripts based on the UTXO model. We employ the pay-to-public-key-hash (P2PKH) script[17]. The transaction script content is illustrated in Figure 3. We apply the Dilithium-Prime post-quantum digital signature scheme[2] within the blockchain's transactions. The signature sizes and security parameters are listed in Table 2, with the security level III parameters selected for this work.

Table 2 Recommended parameter sets for Dilithium-Prime ($n = 251$)

Security params	Security level		
	II	III	V
Modulus q	7,681,537	7,681,537	7,681,537
Matrix dimension (k, l)	(4, 4)	(6, 5)	(8, 7)
Key bound η	2	2	2
Rejection sampling count	3.49	2.96	6.10
Public key size	1,288	1,916	2,544
Private key size	2,504	3,605	4,801
Signature size	2,504	3,233	4 511
LWE hardness	(121, 110)	(162, 146)	(247, 224)
SIS strong hardness	(110, 100)	(169, 153)	(243, 220)
SIS weak hardness	(120, 109)	(184, 167)	(263, 238)

The Dilithium-Prime scheme is a digital signature scheme based on a large-Galois-group prime-degree prime-ideal field, proposed upon the Fiat-Shamir with Aborts technique. Current lattice-based signature schemes, such as CRYSTALS-Dilithium, utilize an underlying algebraic structure of power-of-two cyclotomic rings. Power-of-two cyclotomic rings possess numerous subfields, ring homomorphisms, and a small Galois group. These three characteristics make them susceptible to targeted attacks such as Campbell-Grove-Shepherd, Biasse-Song, Cramer-Ducas-Wesolowski, and S-unit attacks. In contrast, Dilithium-Prime employs the underlying algebraic structure of a large-Galois-group prime-degree prime-ideal field, $\mathbb{Z}_q[x]/(x^n - x - 1)$, where n and q are primes and $x^n - x - 1$ is an irreducible polynomial in $\mathbb{Z}_q[x]$. This structure was introduced by NTRU-Prime [18, 19], a de facto standard used in protocols like OpenSSH. The simplicity of this algebraic structure effectively resists targeted attacks on cyclotomic rings, including Cramer-Ducas-Wesolowski attacks, S-unit attacks, and dual attacks. Furthermore, due to the minimal number of subfields and endomorphisms in the base field, the large-Galois-group prime-degree prime-ideal field resists subfield attacks and endomorphism attacks. Consequently, large-Galois-group prime fields are theoretically considered to provide stronger security guarantees due to their significantly reduced algebraic structure. Moreover, there is a broad consensus on the long-term benefit of eliminating unnecessary algebraic structures in cryptographic system design. For the security advantages of large-Galois-group prime fields over power-of-two cyclotomic rings, one can refer to Refs. [2, 18, 19].

The Dilithium-Prime scheme also incorporates optimizations for efficiency. The public key size was further compressed by using a seed ρ to represent the matrix $\mathbf{A}$ and by employing a series of algorithms to separate and extract the high-order and low-order bits of elements in $\mathbb{Z}_q$. To address the inefficiency of polynomial multiplication in prime fields where the traditional number theoretic transform (NTT) cannot be directly applied, the Dilithium-Prime scheme designed an extended NTT algorithm for these fields. It also introduced a more efficient small polynomial multiplication algorithm tailored for the specific polynomial multiplications required in the signature scheme.

Lattice-based post-quantum digital signature schemes offer high computational efficiency and good scalability. Compared to CRYSTALS-Dilithium, the Dilithium-Prime scheme, by utilizing the large-Galois-group prime-degree prime-ideal field as its underlying structure, provides more robust security. At equivalent security levels, it achieves smaller space overhead and higher signing efficiency, making its characteristics closer to those of current elliptic curve-based digital signature schemes. Therefore, the Dilithium-Prime post-quantum digital signature scheme is applied to transactions in the blockchain. The scheme consists of three parts: the key generation algorithm Dilithium-Prime.KeyGen, the signature algorithm Dilithium-Prime.Sign, and the verification algorithm Dilithium-Prime.Verify. The detailed descriptions are provided in Algorithms 1–3.

Algorithm 1. Key generation algorithm Dilithium-Prime.KeyGen.

Input: security parameter 1^λ;
Output: signature key pair (pk, sk).
$\zeta \leftarrow \{0,1\}^{256}$
$(\rho, \rho', K) \in \{0,1\}^{256} \times \{0,1\}^{512} \times \{0,1\}^{256} := H(\zeta)$
$\boldsymbol{A} \in \mathcal{R}_q^{k\times l} := \mathrm{ExpandA}(\rho)$
$(\boldsymbol{s}_1, \boldsymbol{s}_2) \in \mathcal{S}_\eta^l \times \mathcal{S}_\eta^k := \mathrm{ExpandS}(\rho')$
$\boldsymbol{t} := \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$
$(\boldsymbol{t}_1, \boldsymbol{t}_0) := \mathrm{Power2Round}_q(\boldsymbol{t}, d)$
$tr \in \{0,1\}^{256} := H(\rho \| \boldsymbol{t}_1)$
return $(pk := (\rho, \boldsymbol{t}_1), sk := (\rho, K, tr, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{t}_0))$

Algorithm 2. Signature algorithm Dilithium-Prime.Sign.

Input: private key sk, message M;
Output: signature σ.
$\boldsymbol{A} \in \mathcal{R}_q^{k\times l} := \mathrm{ExpandA}(\rho)$
$\mu \in \{0,1\}^{512} := H(tr \| M)$
$\kappa := 0, (\boldsymbol{z}, \boldsymbol{h}) := \perp$
$\rho' \in \{0,1\}^{512} := H(K \| \mu)$ `// the version with randomization of the algorithm`
    $\rho' \leftarrow \{0,1\}^{512}$
while $(\boldsymbol{z}, \boldsymbol{h}) := \perp$ **do**
    $\boldsymbol{y} \in \widetilde{\mathcal{S}}_{\gamma_1}^l := \mathrm{ExpandMask}(\rho', \kappa)$
    $\boldsymbol{w} := \boldsymbol{A}\boldsymbol{y}$
    $\boldsymbol{w}_1 := \mathrm{HighBits}_q(\boldsymbol{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := H(\mu \| \boldsymbol{w}_1)$
    $c \in \mathcal{B}_\tau := \mathrm{SampleInBall}(\tilde{c})$
    $\boldsymbol{z} := \boldsymbol{y} + c\boldsymbol{s}_1$
    $\boldsymbol{r}_0 := \mathrm{LowBits}_q(\boldsymbol{w} - c\boldsymbol{s}_2, 2\gamma_2)$
    if $\|\boldsymbol{z}\|_\infty \geq \gamma_1 - \beta$ **or** $\|\boldsymbol{r}_0\|_\infty \geq \gamma_2 - \beta$ **then**
        $(\boldsymbol{z}, \boldsymbol{h}) := \perp$
    else
        $\boldsymbol{h} := \mathrm{MakeHint}_q(-c\boldsymbol{t}_0, \boldsymbol{w} - c\boldsymbol{s}_2 + c\boldsymbol{t}_0, 2\gamma_2)$
        if $\|c\boldsymbol{t}_0\|_\infty \geq \gamma_2$ **or** the number of 1 in $\boldsymbol{h}$ is greater than ω **then**
            $(\boldsymbol{z}, \boldsymbol{h}) := \perp$
        end if
    end if
    $\kappa := \kappa + l$
end while
return $\sigma = (\tilde{c}, \boldsymbol{z}, \boldsymbol{h})$

The application process is illustrated through the following example.

When Node A transacts with Node B via transaction Tx, the Dilithium-Prime digital signature algorithm is used. This algorithm does not possess the key derivation functionality inherent to elliptic curve cryptography. In the unlocking script, Node A generates a Dilithium-Prime signature sig_a and writes both the generated signature sig_a and the public key pk_a into the unlocking script. During the process of adding the unlocking script to the transaction, the script's validity is verified. After the transaction is created, it is broadcast to other nodes in the blockchain. Upon receiving the transaction, each node verifies the signature sig_a and adds transaction Tx to the mempool for constructing block templates. Following the execution of the Mul-AuxPoW algorithm, a block is generated. This block is broadcast to other nodes. Upon receiving the block, other nodes verify transaction Tx. After successful verification, they store the block and proceed to the construction and consensus of the next block.

Algorithm 3. Verification algorithm Dilithium-Prime.Verify.

Input: public key pk, message M, signature $\sigma = (\tilde{c}, \boldsymbol{z}, \boldsymbol{h})$;
Output: pass verification (output 1) or fail verification (output 0).

1. $\boldsymbol{A} \in \mathcal{R}_q^{k\times l} := \mathrm{ExpandA}(\rho)$
2. $\mu \in \{0,1\}^{512} := H(H(\rho \| \boldsymbol{t}_1)\|M)$
3. $c := \mathrm{SampleInBall}(\tilde{c})$
4. $\boldsymbol{w}_1' := \mathrm{UseHint}_q(\boldsymbol{h}, \boldsymbol{Az} - c\boldsymbol{t}_1 \cdot 2^d, 2\gamma_2)$
5. **if** $\|\boldsymbol{z}\|_\infty < \gamma_1 - \beta$ and $\tilde{c} = H(\mu \| \boldsymbol{w}_1')$ and the number of 1 in $\boldsymbol{h}$ is less than ω **then**
6. **return** 1
7. **else**
8. **return** 0
9. **end if**

It is worth noting that, as shown in Figure 3, compared to elliptic curve-based signatures like Schnorr signatures[20], the signature length increases significantly, leading to a longer transaction script. To maintain efficient transaction processing per block, it is recommended to increase the block size capacity.

3 Theoretical Analysis

3.1 Security analysis of Mul-AuxPoW

The Mul-AuxPoW consensus mechanism expands the sources of computational power beyond the base PoW, allowing any blockchain using either the Sha256 or Scrypt hash function in its consensus to act as a parent chain and contribute computational power. The analysis of the consensus mechanism will examine its behavior under common attacks such as Sybil attacks and double-spending attacks.

(1) 51% computational power attack

A 51% computational power attack[21] involves controlling at least 51% of the network's computational power to gain control of the blockchain network and subsequently alter transactions. In the Mul-AuxPoW consensus mechanism, the sources of computational power include the child chain's own power and that of other parent chains providing auxiliary consensus. Compared to the standard PoW consensus, this significantly diversifies the computational power sources, making it considerably more difficult for an attacker to amass over 51% of the total power. The resistance of the PoW consensus mechanism to 51% computational power attacks has already been proven[22].

(2) Double-spending attack

A double-spending attack occurs when an attacker spends the same cryptocurrency tokens more than once. A common method is to perform a double-spend via selfish mining: forking the blockchain, mining blocks without broadcasting them, and then releasing a branch longer than the main chain to double-spend the tokens from the transactions in that branch. In the Mul-AuxPoW consensus mechanism, the increased diversity of computational power sources means an attacker would need to control significantly more power to outperform the main network, thereby reducing the likelihood of successful forking. Furthermore, similar to Bitcoin, mechanisms like UTXO, timestamps, and requiring multiple confirmations are employed to resist double-spending attacks.

(3) Malicious information modification during data interaction

The Mul-AuxPoW consensus process primarily involves two types of interactions: between parent chains and the mining pool and between the child chain and the mining pool. During these interactions, the mining pool submits the child chain's block header hash and difficulty value to the parent chain, and the parent chain returns the computation result to the child chain via the mining pool. If the information submitted to the parent chain is maliciously altered, it will only cause the parent chain's auxiliary consensus attempt to fail. The parent chain's computational power continues to secure its own consensus, and no resources are wasted. When the computation result is returned from the parent chain to the child chain, the child chain first verifies that the child chain's block header hash and difficulty value match those written into the parent chain. For an attacker to produce a valid hash result after maliciously modifying the data, they would need computational power equivalent to that currently participating in the parent chain's consensus.

3.2 Security analysis of DAA

For blockchains like Bitcoin that employ a DAA, two common computational power attacks are prevalent. One is the pool-hopping attack, where an attacker repeatedly decreases computational power across multiple cycles. This increases the overall block production time, leading to a subsequent decrease in difficulty. The attacker then increases computational power in the next cycle. By leveraging this method, the attacker can shift computational power between different public blockchains, directing it to a public blockchain that has previously experienced a power reduction and thus has a lower difficulty. This makes it easier to obtain extra profits. Subsequently, the attacker hops computational power to other public blockchains using the same strategy, gaining excessive profits while significantly reducing the block production time in that cycle and undermining stability. The other attack is the reverse: it repeatedly increases computational power across multiple cycles and reduces the overall block production time, thereby increasing the difficulty and then drastically reducing power in the next cycle. This severely increases the block production time and can, in extreme cases, halt block production entirely. While this method is unlikely to generate direct profit for the attacker, it can seriously impact the blockchain's stability.

In the Mul-AuxPoW DAA, two types of parent chains provide computational power sources. These two attacks can be further categorized based on whether parent chain A with larger computational power, or parent chain B with relatively smaller computational power, performs sudden computational power surges or drops. The following analyzes the four possible attack scenarios.

(1) Parent chain A with larger computational power continuously reduces and then suddenly increases computational power. The main goal of this attack is for parent chain A to gain more profit. However, the DAA aims to maintain a specific block production ratio between parent chains A and B, with recommended parameters generally setting parent chain A's block production ratio much higher than parent chain B's. In this attack, continuously reducing computational power would persistently decrease previous block production rewards, while only yielding relatively low extra profits in one cycle.

(2) Parent chain A with larger computational power continuously increases and then suddenly reduces its computational power. Parent chain A continuously increasing computational power raises its number of blocks produced and reduces the overall block production time; both outcomes require increasing the difficulty for parent chain A. To prevent this attack, when these situations are detected, particularly when parent chain B's number of blocks produced falls below the parameter $B_{\mathrm{low}_{\mathrm{limit}}}$ but not because parent chain B itself reduced computational power to zero, the algorithm significantly reduces the magnitude of the difficulty increase for parent chain B.

(3) Parent chain B with relatively smaller computational power continuously reduces and then suddenly increases computational power. This attack aims for parent chain B to continuously reduce its number of blocks produced, failing to maintain the target block production ratio, which continuously lowers the difficulty. A sudden subsequent computational power increase would then significantly increase extra rewards. To prevent this attack, when parent chain B's block production continuously decreases, and the block production time increases, the algorithm controls the extent of the difficulty decrease for parent chain B, thereby continuously reducing the potential profit from such an attack.

(4) Parent chain B with relatively smaller computational power continuously increases and then suddenly reduces its computational power. Since parent chain A accounts for the majority of the number of blocks produced and has a greater impact on block production time, this quantitative attack by parent chain B is ineffective. Conversely, it would significantly reduce parent chain B's own rewards.

3.3 Correctness analysis of Mul-AuxPoW

As described in Section 4.2, the Mul-AuxPoW consensus mechanism constructs block templates with different difficulty targets for the two types of parent chains prior to consensus. Based on the parent chain type, the corresponding difficulty D and the child chain's block header hash value H are written into the parent chain's CoinBase transaction. The CoinBase transaction is part of the block's transactions. Utilizing the properties of the Merkle tree, if the content of the CoinBase is modified, the resulting Merkle root will also change, consequently altering the parent chain's block header hash. In the original PoW consensus mechanism used by the parent chain, the formula is

$$\mathit{Hash}(\mathit{nonce} + \mathit{hashBlockHeader}) < \mathit{Target}$$

If the *hashBlockHeader* value changes, the result of the iterated hash computation will differ. Therefore, H as part of the parent chain's CoinBase transaction, has a one-to-one correspondence with the parent chain's *hashBlockHeader*. Since H is part of the child chain's PoW, it can be understood that the corresponding difficulty D is effectively being computed within the parent chain's PoW consensus, thereby satisfying the PoW requirement.

The parent chain's block template that meets difficulty D is also stored in the new structural section of the child chain block. The specific verification process is detailed in Stage 4 of Section 2.2. It leverages the properties of the Merkle tree to verify the presence of H in the CoinBase transaction. Then, using the Merkle branch containing the CoinBase, it employs the simplified payment verification method to verify the correctness of the Merkle root value and the block header hash. Finally, it verifies whether the PoW meets the required difficulty. This means the child chain's PoW computation satisfies the difficulty requirement, resulting in successful block production.

3.4 Correctness analysis of DAA

DAA can be broadly divided into four parts. The first part aims to stabilize the block production time. It calculates $\frac{AT}{ET}$, the ratio of the actual block production time for φ blocks on the child chain to the expected block production time and segments the formula into three ranges based on this value's magnitude. If it is greater than 1.05, it indicates the actual block production time is slightly longer than the target, so the overall difficulty needs to be increased to reduce the total time. If it is between 0.95 and 1.05, it indicates the actual block production time is roughly close to the target, making balancing the block production ratio more critical at this stage. If it is less than 0.95, it indicates the actual block production time is slightly shorter than the target, so the overall difficulty needs to be decreased to increase the total time.

The second part aims to stabilize the block production ratio between the two types of parent chains, calculating two values. For parent chain A with relatively larger computational power, the first thing is to calculate $\alpha \cdot \dfrac{y_n}{x_n}$, comparing the ratio of blocks produced by parent chain B with relatively small computational power to parent chain A with relatively larger computational power against the target block production ratio α. If the product is 1, it means the actual ratio matches the target, requiring no extra difficulty adjustment. If it is greater than 1, it means parent chain B's number of blocks produced has increased, requiring an additional decrease in difficulty for parent chain A and an increase for parent chain B, but the change value must be bounded within $[\mu, \xi]$. However, since this value is based on a ratio, the degree of change increases more significantly when the actual ratio deviates more from the target. Therefore, for parent chain A, a second value is calculated: the ratio of the target blocks generated per cycle φ to the number of blocks produced by parent chain A, which is then compared to the target block production ratio. The calculation is analogous for parent chain B. The two values are weighted and averaged, resulting in a value that represents the comprehensive calculation of the relationship coefficient between the block production ratios of the two types of parent chains and the target value. The proportion coefficients are β and γ.

The third part calculates T_{A_n} and T_{B_n}. For parent chain A, the exponent $\dfrac{x_{n-1} - x_n}{x_A}$ part represents the ratio of the difference in the number of blocks produced between cycle $n-1$ and cycle n to the weighted average of the change in the number of blocks produced for A and B over recent cycles. When the number of blocks produced in this stage is more than in the previous stage, the difficulty will appropriately become greater; conversely, the difficulty will become smaller. Moreover, the degree of change will be limited and influenced by the change in the number of blocks produced in recent stages. If the change degree in recent stages is small, the change in the latest stage is more likely due to network instability, so this parameter will be more pronounced than when the change degree in recent stages is large. If the change degree in recent stages is large, the change in the latest stage might just be because the difficulty is adjusting to a stable state with large fluctuations, so the influence of this parameter should be reduced, the same as x_A. The base δ_T is a parameter.

The second part and the third part serve as adjustment coefficients, controlling the block production ratio and controlling the degree of block production fluctuation, respectively. The product of the two parts needs to be controlled according to the ratio size calculated in the first part. This is because, assuming the first part value is greater than 1.05, it means that to control and decrease the block production time, the difficulty needs to be decreased, and the product of the second and third parts cannot be less than 1, or otherwise it may cause the product of the three parts to be less than 1, finally increasing the difficulty; if the first part value is between 0.95 and 1.05, it means the time is basically stable, and the focus needs to be on controlling the parent chains' block production ratio and controlling block production fluctuation, so no boundary needs to be set; if the first part value is less than 0.95, it means that to control and increase the block production time, the difficulty needs to be increased, and the product of the second and third parts cannot be greater than 1, or otherwise it may cause the product of the three parts to be greater than 1, finally decreasing the difficulty. This also indicates that during difficulty adjustment, the first need is to control the block production time to be as stable as possible and then ensure the stability of the block production ratio and fluctuation, which aligns with the concept of DAAs like Bitcoin's.

The fourth part sets some special cases based on the algorithm characteristics. When

$y_n \leq B_{\mathrm{low}_{\mathrm{limit}}}$, based on this condition alone, theoretically the difficulty for parent chain B should be decreased. There are two situations where $y_n \leq B_{\mathrm{low}_{\mathrm{limit}}}$ occurs: first, a sudden increase in computational power from parent chain A (or a decrease in parent chain A's difficulty in the previous stage). For this situation, the difficulty for parent chain A should be increased, and the difficulty for parent chain B should be decreased but not too much. Second, parent chain B launches a sudden computational power decrease attack, reducing its blocks in this stage, aiming to decrease parent chain B's difficulty in the next stage or increase parent chain A's difficulty, making parent chain B produce more blocks in the next stage and get more profit. This situation is the most likely to occur, and the extent of the difficulty decrease for parent chain B should be reduced, but it will not increase or maintain parent chain B's difficulty, to prevent other situations. But when $c_{\mathrm{less}} \geq 1$, it can be judged that parent chain B is launching multiple rounds of sudden computational power decrease attacks. If not, during the first adjustment, parent chain A's difficulty would increase, and parent chain B's difficulty would slightly decrease. If parent chain B's number of blocks produced still does not reach above 5 blocks, it indicates parent chain B's computational power has been in an unhealthy state for multiple stages, and the extent of its difficulty decrease is reduced again, making the profit from parent chain B launching attacks less and the cost greater, thus ensuring system security. For the situation where parent chain A's difficulty is still increased when this occurs, this is unfair to parent chain A: firstly, this setup makes it so that even if parent chain B's attack succeeds, the block production time per block is still prolonged, and the profit per unit time is still not optimistic; secondly, parent chain A has a supervisory role; miners are profit-driven, and if this situation occurs, they will flock to parent chain B for mining, then in the next cycle, the difficulty will be adjusted to an appropriate size; moreover, it prevents the possibility of the first situation occurring. Other situations ensue: parent chains A and B simultaneously withdraw part of their computational power, but parent chain B withdraws more; parent chain B's difficulty increases significantly in the previous stage, leading to fewer blocks in this round; or other probabilistic reasons lead to fewer blocks. For this kind of situation, we will slowly decrease the computational power. After one adjustment round, parent chain B's number of blocks produced will exceed $B_{\mathrm{low}_{\mathrm{limit}}}$, and the difficulty for parent chain B will be quickly adjusted according to other corresponding formulas. The situation for $y_n \geq A_{\mathrm{low}_{\mathrm{limit}}}$ is similar.

3.5 Analysis of Mul-AuxPoW consensus mechanism and transaction efficiency

Regarding consensus time, the time consists of the block template construction time t_v, consensus hash computation time t_c, block propagation time t_p, and block verification time t_y by nodes. Compared to the PoW consensus mechanism, the Mul-AuxPoW consensus mechanism primarily increases t_v and t_y. During block template construction, interaction with parent chains and the mining pool is required to build the block template; during block verification, the AuxPoW structure needs to be verified. However, in the consensus process, the hash computation time t_c has a profound impact on the total time, while the other times are comparatively negligible. Therefore, the overall consensus time increases, but the added time is negligible by comparison. It should be noted that within t_v and t_y, verifying transaction signatures is required; the time needed depends on the number of signatures and the single verification time. Here, the focus is primarily on the impact of the consensus structure on time.

Regarding space, a 32-byte AuxPoW hash pointer is added to the block header. An additional *CAuxPoW* structure is included, containing a 32-byte auxiliary block parent chain block header hash, a 144-byte auxiliary block parent chain CoinBase transaction, and a 0–576 byte auxiliary block parent chain CoinBase transaction Merkle branch. Overall, this increases the space by 176–752 bytes. Compared to Bitcoin's 1 MB block size, this represents a total increase of approximately 0.03% in space. Furthermore, for the blockchain applying post-quantum signatures, if the security level III parameters from Table 2 are selected, the signature size increases to 3,233 bytes and the public key size to 1,916 bytes, significantly increasing the transaction space. To improve blockchain efficiency, the block size is expanded to 30 MB. Compared with the application of adding the Mul-AuxPoW consensus mechanism, the consensus mechanism has increased the space by 0.001%.

The impact of post-quantum signatures on transactions is detailed below. Regarding time, since transactions are generated via scripts, and the main operations in the script involve generating signatures, filling in public keys and public key hashes, opcodes, etc., the signing time primarily affects the transaction. Due to the application of the Dilithium-Prime signature, the overall transaction generation time becomes longer.

Regarding space, by taking the P2PKH script shown in Figure 3 as an example, the unlocking script consists of a 3,234-byte signature and a 1,916-byte public key. The locking script consists of opcodes and a 160-byte public key hash. Compared to those in the locking script, the signature and public key in the unlocking script contribute more significantly to the space overhead. Relative to the time impact, applying the Dilithium-Prime signature has a greater impact on space, significantly increasing the size of a single transaction.

3.6 Comparative analysis of common PoW public blockchains

The blockchain system designed in this paper is implemented based on blockchains utilizing the UTXO script architecture and the PoW consensus algorithm, represented by public blockchains like Bitcoin and Litecoin. A functional comparison is shown in Table 3.

Table 3 Function comparison with other public blockchains

Blockchain	Consensus algorithm	DAA	Signature algorithm
Bitcoin	PoW consensus algorithm	Adjust difficulty based on the ratio of block production time to target time	ECDSA
Litecoin	AuxPoW consensus algorithm	Adjust difficulty based on the ratio of block production time to target time	ECDSA
Blockchain in this paper	Mul-AuxPoW consensus algorithm	Adjust difficulty based on the ratio of block production time to target time, the block production ratio of parent chains, and the stability of the number of blocks produced by parent chains over multiple cycles	Post-quantum secure Dilithium-Prime algorithm

In terms of the consensus algorithm, Bitcoin uses the PoW consensus algorithm with the Sha256 hash function, and its computational power source comes solely from nodes that choose to participate in Bitcoin's consensus. Litecoin and Dogecoin use the auxiliary proof-of-work (AuxPoW) consensus algorithm with the Scrypt hash function. Their computational power sources include nodes participating in the consensus of these two public blockchains. The public blockchain designed in this paper employs the Mul-AuxPoW consensus algorithm. Building upon the PoW consensus algorithm, this mechanism leverages the properties of hash functions to allow nodes from public blockchains using either the Sha256 or Scrypt hash function with the PoW consensus algorithm to provide computational power support. This significantly diversified the sources of computational power, thereby enhancing the blockchain's security and the utilization efficiency of computational power.

In terms of difficulty adjustment, Bitcoin and Litecoin adjust the difficulty based on the ratio of the block production time to the target time within a certain period, reflecting changes in computational power. The public blockchain designed in this paper, in addition to this, incorporated factors for the stability of the number of blocks produced by parent chains over multiple cycles and the ratio of the number of blocks produced between the two types of parent chains to adjust the difficulty. This allowes it to better adapt to the Mul-AuxPoW consensus algorithm and fluctuations in computational power. To counter computational power attacks, special-case difficulty adjustment formulas are also designed, improving the stability of the system's block production.

In terms of signature algorithm, both Bitcoin and Litecoin use ECDSA. With the continuous advancements in quantum computers, their security is severely challenged. The public blockchain designed in this paper adopts the post-quantum secure Dilithium-Prime algorithm, adapted for the UTXO script design. It meets standard functional requirements while significantly enhancing the security of system transactions.

4 Experimental Verification

We implement a blockchain system in C++ that incorporates the Mul-AuxPoW consensus mechanism, DAA, and the application of the Dilithium-Prime post-quantum signature algorithm in transactions. Five cloud servers, each with 4 cores and 16 GB of memory, are set up for simulation experiments. Furthermore, by using Python scripts and Docker technology, 30 blockchain nodes are uniformly constructed on each cloud server to set up the block nodes. The relevant experimental parameters are set as follows.

(1) The first parameter set from Table 2 is selected for the DAA, with a relatively small cycle length to facilitate system testing.

(2) Based on the current real-world computational power environment, resources were allocated as follows: 140 nodes for the parent chain running the Sha256 hash function, 5 nodes for the parent chain running the Scrypt hash function, and 5 nodes for the child chain.

(3) The mining pool interaction process was simulated using C++. The network communication latency between cloud server nodes was approximately 0.6 ms, with a bandwidth of 3 Mb/s.

In this section, we conduct experiments on the efficiency of the Mul-AuxPoW consensus mechanism, simulated attacks against DAA, and the transaction efficiency with the applied post-quantum signatures, to test the system's stability and efficiency.

4.1 Mul-AuxPoW consensus mechanism efficiency

(1) Mul-AuxPoW consensus mechanism speed

In this experiment, we simulate the block consensus process, performing 1,000 consensus rounds (excluding transactions), and we calculate the average time and proportion for each process stage. The experimental results are shown in Table 4. It can be observed that the consensus hash computation time t_c constitutes the vast majority of the total consensus process time. In comparison, the block template construction time t_v (without transactions) and the block verification time t_y are negligible.

Table 4 Average time and proportion of block consensus process stages

Process	t_v	t_c	t_p	t_y
Time (ms)	10.52	120,000.00	0.21	0.58
Proportion (%)	<0.01	99.99	<0.01	<0.01

(2) Mul-AuxPoW consensus mechanism storage

In the experiment, we tested the boundary values of block size, namely the block header length, *CAuxPoW* module size, and the overall block size for blocks containing only a CoinBase transaction versus blocks utilizing the full space capacity. The experimental results are shown in Table 5 and Table 6, respectively. It can be observed that for blocks containing only a CoinBase transaction, the block header and the *CAuxPoW* consensus structure occupy the main space. As the number of transactions increases until the block structure is fully utilized, the block transactions occupy the primary space, and the additional space overhead of the *CAuxPoW* consensus and the block header becomes negligible.

Table 5 Space overhead and proportion for blocks containing only CoinBase transaction

Process	Block header	*CAuxPoW*	Head
Space (bytes)	80	176	301
Proportion in block (%)	26.58	58.47	100

Table 6 Space overhead and proportion for fully utilized blocks

Process	Block header	*CAuxPoW*	Head
Space (bytes)	80	752	30,000,000
Proportion in block (%)	<0.01	<0.01	100

4.2 DAA attack test

In this experiment, we simulate the stability of the blockchain when the two types of parent chains perform computational power attacks. Computing power is measured in hashrate, which refers to the number of hash calculations performed per second. 1 kH/s and 1 MH/s represent 1,000 and 1,000,000 hash operations per second, respectively. First, by keeping the computational power of the parent chain B (using the Scrypt hash function) stable, the computational power of the parent chain A (using the Sha256 hash function) is sequentially subjected to a significant increase, a significant decrease, a slight increase, and a slight decrease over a period of cycles. The overall block production time of the child chain system is shown in Figure 4. It can be observed that the overall block production time initially decreases and then increases following the changes in computational power. Depending on the magnitude of the change, it can quickly adjust back to the normal fluctuation range, stabilizing the block production time within 2–4 cycles. The number of blocks produced by the two parent chain types is shown in Figure 5. After the number of blocks produced by the parent chain A changes drastically with the computational power variation, it can stabilize around the target value within 2–4 cycles.

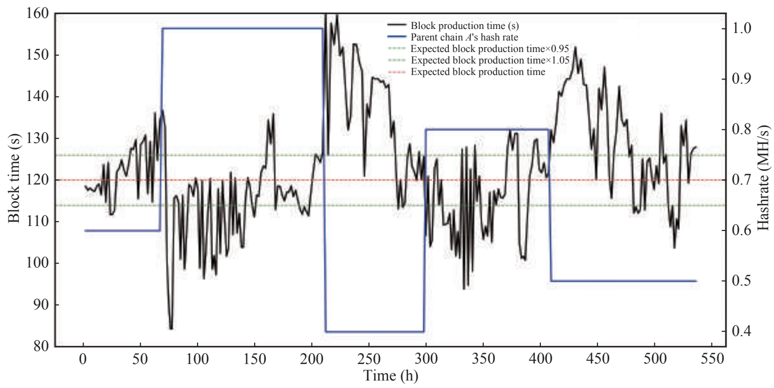

Figure 4 System's block production time under computational power attack from the parent chain A

Similarly, by keeping the computational power of the parent chain A (using the Sha256 hash function) stable, the computational power of the parent chain B (using the Scrypt hash function) is sequentially subjected to a significant increase, a significant decrease, a slight increase, and a slight decrease over a period of cycles. The overall block production time of the child chain system is shown in Figure 6, and the number of blocks produced by the two parent chain types is shown in Figure 7.

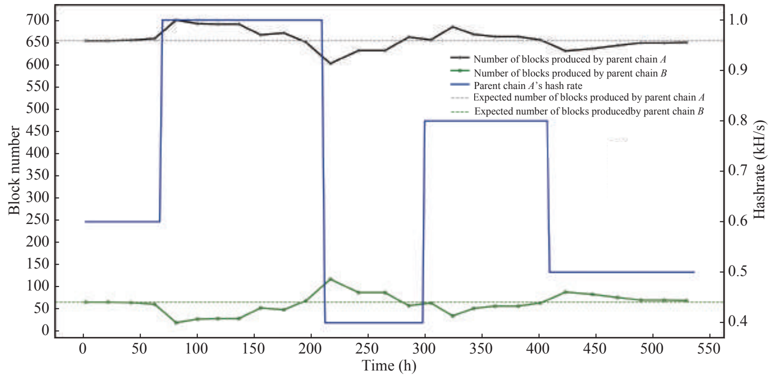

Figure 5 Number of blocks produced by the two types of parent chains under computational power attack from the parent chain A

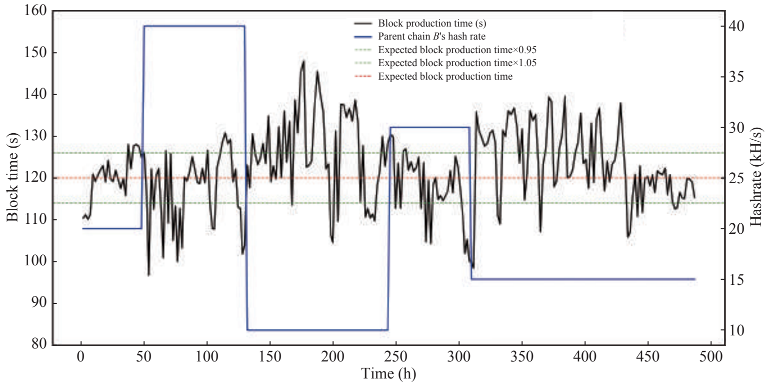

Figure 6 System's block production time under computational power attack from parent chain B

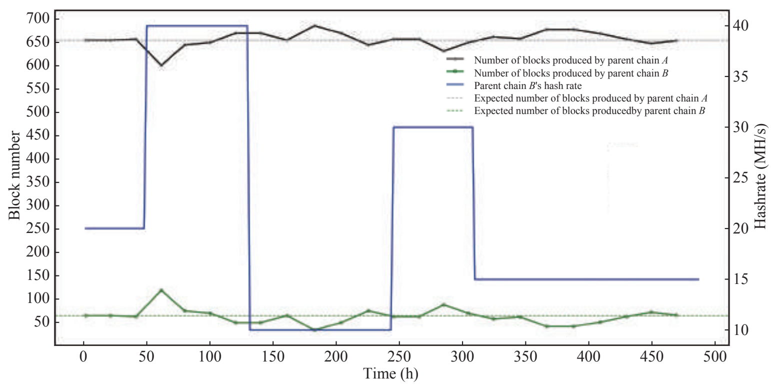

Figure 7 Number of blocks produced by the two types of parent chains under computational power attack from the parent chain B

4.3 Experimental verification of transaction efficiency with post-quantum signatures

(1) Transaction speed analysis with post-quantum signatures

In this experiment, we simulate the transaction process, performing 1,000 transactions, and calculate the average time and proportion for signing and the overall transaction. The experimental results are shown in Table 7. It can be observed that the average time for a single signature is 0.5 ms, accounting for 28.8% of the total time for one transaction. The application of the post-quantum signature Dilithium-Prime has a relatively minor impact on transaction time. Additionally, transactions involve other time-consuming operations such as reading from and writing to memory.

Table 7 Average time and proportion for signing and transactions

Process	Signing	Transaction
Time (ms)	0.55	19.09
Proportion (%)	28.8	100

(2) Transaction storage overhead with post-quantum signatures

Dilithium-Prime has a significant impact on transactions in terms of space. In this experiment, we test the spatial storage overhead of transactions and signatures by simulating transactions. The experimental results are shown in Table 8. Within a transaction, the signature and public key sizes occupy the majority of the transaction space, while opcodes and the public key hash also occupy a certain amount of space. It can be concluded that the application of the post-quantum signature Dilithium-Prime has a significant impact on the spatial overhead of transactions.

Table 8 Space overhead and proportion for signatures and transactions

Process	Signing	Public key	Transaction
Space (bytes)	3,233	1,916	5,374
Proportion (%)	60.2	35.7	100

5 Conclusion

We designed and implemented a post-quantum blockchain system based on the Mul-AuxPoW consensus mechanism and a post-quantum signature algorithm utilizing a large-Galois-group prime-degree prime-ideal field. The system employes the Mul-AuxPoW consensus mechanism. Through a child chain, mining pool, and parent chain tripartite interaction model, the algorithm enables blockchains using the Sha256 and Scrypt hash functions in their consensus mechanisms to act as parent chains and provide auxiliary consensus for the child chain. This consensus mechanism increases the child chain's sources of computational power and improves the utilization rate of computational power from existing mining pools and miners. For this consensus mechanism, DAA was designed to simultaneously stabilize the block production ratio of the two parent chain types and the block production time, and it can resist attacks involving sudden surges or drops in computational power. For transactions, Dilithium-Prime, based on a large-Galois-group prime-degree prime-ideal field, was adopted, which significantly reduces the algebraic structure of the underlying mathematical problem upon which the post-quantum signature algorithm is based, thereby enhancing the post-quantum security and reliability of transactions. Finally, theoretical analysis and experimental verification of the system were provided.

References

[1] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf

[2] Dong YF, Fang BY, Liang ZC, Zhao YL. Efficient lattice-based digital signature scheme in large-Galois-group prime-degree prime-ideal field. Ruan Jian Xue Bao/Journal of Software, 2025, 36(2): 776–804 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/ 7164.htm [doi: 10.13328/j.cnki.jos.007164]

[3] Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Review, 1999, 41(2): 303–332. [doi: 10.1137/S0036144598347011]

[4] National Institute of Standards and Technology. PQC standardization process: Announcing four candidates to be standardized, plus fourth round candidates. 2022. https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4

[5] Bai S, Ducas L, Kiltz E, *et al*. CRYSTALS-Dilithium algorithm specifications and supporting documentation (Version 3.1). 2023. https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

[6] Fouque PA, Hoffstein J, Kirchner P, Lyubashevsky V, Pornin T, Prest T, Ricosset T, Seiler G, Whyte W, Zhang ZF. Falcon: Fast-Fourier lattice-based compact signatures over NTRU (Specification v1.2). 2020. https://falcon-sign.info/falcon.pdf

[7] Torres WAA, Steinfeld R, Sakzad A, Liu JK, Kuchta V, Bhattacharjee N, Au MH, Cheng J. Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (Lattice RingCT v1.0). Proc. of the 23rd Australasian Conf. on Information Security and Privacy. Wollongong: Springer, 2018. 558–576. [doi: 10.1007/978-3-319-93638-3_32]

[8] AyakaEna. Whitepaper draft. 2022. https://github.com/ArielCoinOrg/docs/blob/main/whitepaper.md

[9] HCASH Foundation. Hcash technical yellow paper. 2018. https://h.cash/themes/zh-cn/images/YellowPaper-1.01-Chinese-1.pdf

[10] Ding JT. A new proof of work for blockchain based on random multivariate quadratic equations. In: Applied Cryptography and Network Security Workshops: ACNS 2019 Satellite Workshops. Bogota: Springer, 2019. 97–107. [doi: 10.1007/978-3-030-29729-9_5]

[11] Jackalys. Quantum Resistant Ledger (QRL). 2016. https://github.com/theQRL/Whitepaper/blob/master/QRL_white paper.pdf

[12] Zweil M. Mochimo: Post-quantum currency. 2018. https://mochimo.org/assets/files/mochimo_wp_EN.pdf

[13] Liu YZ, Liu JW, Zhang ZY, Xu TG, Yu H. Overview on blockchain consensus mechanisms. Journal of Cryptologic Research, 2019, 6(4): 395–432 (in Chinese with English abstract). [doi: 10.13868/j.cnki.jcr.000311]

[14] Judmayer A, Zamyatin A, Stifter N, Voyiatzis AG, Weippl E. Merged mining: Curse or cure? In: Data Privacy Management, Cryptocurrencies and Blockchain Technology: ESORICS 2017 Int'l Workshops, DPM 2017 and CBT 2017. Oslo: Springer, 2017. 316–333. [doi: 10.1007/978-3-319-67816-0_18]

[15] Noda S, Okumura K, Hashimoto Y. An economic analysis of difficulty adjustment algorithms in proof-of-work blockchain systems. 2022. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3410460

[16] Cortesi E, Bruschi F, Secci S, Taktak S. A new approach for Bitcoin pool-hopping detection. Computer Networks, 2022, 205: 108758. [doi: 10.1016/j.comnet.2021.108758]

[17] Bistarelli S, Mercanti I, Santini F. An analysis of non-standard bitcoin transactions. Proc. of the 2018 Crypto Valley Conf. on Blockchain Technology (CVCBT 2018). Zug: IEEE, 2018. 93–96. [doi: 10.1109/CVCBT.2018.00016]

[18] Bernstein DJ, Chuengsatiansup C, Lange T, Van Vredendaal C. NTRU prime: Reducing attack surface at low cost. Proc. of the 24th Int'l Conf. on Selected Areas in Cryptography (SAC 2017). Ottawa: Springer, 2018. 235–260. [doi: 10.1007/978-3-319-72565-9_12]

[19] Bernstein DJ, Brumley B, Chen MS, *et al*. NTRU Prime: Round 3. 2023. https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

[20] Schnorr CP. Efficient signature generation by smart cards. Journal of Cryptology, 1991, 4(3): 161–174. [doi: 10.1007/BF00196725]

[21] Aponte-Novoa FA, Orozco ALS, Villanueva-Polanco R, Wightman P. The 51% attack on blockchains: A mining behavior study. IEEE Access, 2021, 9: 140549–140564. [doi: 10.1109/ACCESS.2021.

3119291]

[22] Tian GH, Hu YH, Chen XF. Research progress on attack and defense techniques in block-chain system. Ruan Jian Xue Bao/Journal of Software, 2021, 32(5): 1495–1525 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6213.htm [doi: 10.13328/j.cnki.jos.006213]

Yutian Wang, Ph.D. candidate. His research interest is blockchain.

Yifan Dong, Ph.D. candidate. Her research interest is lattice cryptography.

Shurui Jia, master's degree candidate. Her research interest is blockchain.

Yafang Yang, Ph.D. Her research interests include cryptography and its applications.

Mingxi Chen, master's degree candidate. His research interest is blockchain.