

Ethereum Security Based on Blockchain Framework and its Algorithms and Components

Zhijian Liu ^{1,†}, Zihan Shen ^{2,†}, Hongfei Wang ^{3,*}, Qianjia Zou ^{2,†}

¹ Department of Computer Science, College of Mobile Telecommunications Chongqing University of Posts and Telecom, Chongqing, China

² Department of Computer Science, Southwest Jiaotong University, Chengdu, China

³ Department of Computer Science, California state university, Fullerton, Shenyang, China

* Corresponding Author Email: wanghf0914@csu.fullerton.edu

† These authors contributed equally.

Abstract. With the increasing amount and trading volume, account security has become an issue that people have to consider. The security of Ethereum is discussed in four different aspects. This paper reviews the application of cryptography in Ethereum, including the importance of relevant hash algorithms and digital signature techniques for securing data. The basic structure of the Merkle Tree and the role of its modified data structure, performed in the security mechanism of Ethereum are also analyzed. This paper also analyzes the related Merkle Proof algorithm. Additionally, the definition and working mechanism of Gas in Ethereum are also provided, through which the operating mechanism and creation method of Gas can guarantee the security of Ethereum's processing power. Finally, this paper indicates the underlying vulnerabilities and possible attacks on Bitcoin and Ethereum, including double spending attacks under proof of work and further introduces the related solutions. This paper can provide researchers good references on Ethereum security problem.

Keywords: Block chain, merkle patricia tree, Gas, block size, ethash, proof-of-work, double spending

1. Introduction

In recent years, virtual currency has gradually entered everyone's field of vision. Ethereum is the second largest platform among many virtual currency trading platforms, and due to its decentralized characteristics without central authority control, many security risks have appeared in Ethereum. Blockchain is a technology that enables a centralless distributed ledger. The basic unit of the general ledger technology is the transaction, and the whole ledger is composed of a strip of transactions, with a number of transactions recorded in each block, connecting the blocks to form a complete ledger. The benefit of such an organized ledger is that individual transactions on the chain cannot be tampered based on cryptographic algorithms. The greatest benefit is that the data is easy to keep intact and is more secure to some extent. However, this would be more likely to lead to a rapid growth of data, and there is a problem of data overload for blockchain systems of enterprise-level applications. To solve this problem, the Merkle Tree structure has been introduced in the ETH. Merkle Patricia Tree, a modified data structure that combines the advantages of both Merkle Tree and Prefix tree, is an important data structure used in Ether to organize and manage account data and generate transaction set hashes. The MPT tree provides a proof method called Merkle Proof for light node scaling to achieve simple payment verification. Gas is a function of fuel in Ethereum. Gas, an unit of measurement for the computational effort, is required to perform various activities on the Ethereum network. Since each Ethereum transaction uses computational resources, each transaction is charged. One of the most important functions of Gas is to protect the security of Ethereum's computing power and avoid many redundant malicious operations by charging for each performed activity. The Ethereum network also has a double-spending attack under the mechanism of the blockchain platform.

Under the proof-of-work mechanism, double-spending attacks can be effectively reduced. However, 51 hashrate attacks are still unavoidable.

2. Security Based On The Ethereum Blockchain

2.1. Security in Ethereum

(1) Ethereum

Ethereum is born on the basis of Bitcoin. They are all equivalent to a distributed system, and Ethereum has its own peer-to-peer(P2P) protocol, so that its products can be transferred freely and safely. Ethereum uses the Ethereum Virtual Machine (EVM) to build applications based on simple and modular rules. This application is called a contract and allows users to write some operations according to their own ideas. Many people claim that users and Ethereum form a computer for use by people all over the world. The blockchain is the ROM of the computer, and smart contracts are simply programs stored on it. The Ethereum miners have computation power and act as CPU.

(2) Security of the transaction process

The transaction process goes through submitting a transaction application, private key signing transaction, submitting transaction, verifying transaction, and broadcasting transaction. Due to the issue of fund security, this process may not be safe. For example, someone may impersonate the user. This can be solved by cryptography. After generating a 1-2256 random number offline and getting a 256-bit string through the hash algorithm (SHA256), the secret key can be obtained. The fewer people know, the safer the key is. After that, the public key can be obtained through the multiplication of the elliptic curve equation [1, 2] as presented in Formula 1 as below [3]:

$$K=k \times G \quad (1)$$

Where k is the private key, G is the constant point (base point), and K is the public key we calculated. Since the elliptic curve equation is a one-way function, it is very difficult to find k through G if K is known, and it will be very simple if the holder of the private key calculates the public key. First $r = x_1 \bmod n$ is calculated, and n is the multiplicative order of the elliptic curve. If $n = 0$, it needs to recalculate $k * G$, and then calculate $s = k^{-1}(z + rK) \bmod n$. If $s = 0$, it also needs to recalculate $k * G$, and finally gets (r, s) as a number signed to it. After each transmission of information, k is updated in time, otherwise the private key can be obtained through formula conversion [3]. The transaction can be signed when the transaction is initiated, so that it can prove that the key holder, instead of an impersonator, can make the process more secure.

2.2. Blockchain security

According to the definition and workflow, blockchain technology is a trust mechanism. Consensus mechanisms and encryption algorithms are key technologies for blockchain to ensure data security, immutability, and transparency. Among them, the consensus mechanism mainly solves the problems of building the block and maintaining the unity of the blockchain. Encryption algorithms are used to solve the problem of ownership of electronic money. The exploration of blockchain technology in different fields mainly lies in the selection of appropriate consensus mechanisms and encryption algorithms. The consensus mechanism ensures that the node data on the blockchain is consistent. The main mechanisms include Proof of Work (PoW), Proof of Stake (PoS), Delegate Proof of Stake (Dpos), and Practical Byzantine Fault Tolerance (PBFT). Information encryption is a key link in the blockchain, mainly hash functions and asymmetric encryption algorithms. Among them, the asymmetric encryption part uses the private key to prove the ownership of the node, which is completed through a digital signature. It uses a hash algorithm to convert an input of any length into a fixed-length output composed of letters and numbers. The output is irreversible and can not be tampered [4].

3. Merkle Tree In Ethereum

3.1. Merkle Tree

(1) Analysis of Merkle Tree Structure in Blockchain

Merkle tree (Hash Tree), first proposed by the computer scientist Ralph Merkle, is a kind of data structure that stores hash values. The merkle tree is regarded as a hash list, and it introduces a tree structure on this basis, which leads to increased flexibility. To be more specific, the tree can be built from bottom up. At the bottom of the merkle tree, data can be divided into small data blocks, with the corresponding hash values. When the leaf node reaches the upper structure, two adjacent hash values are merged into a string. The process is repeated and the merkle tree is finally formed, with only one root node on the top of the tree.

(2) Merkle Patricia Tree in Ethereum (MPT)

According to the account model used in the blockchains [5], huge storage space is more likely to be consumed during the process. Therefore, the MPT is introduced in Ethereum, which is an improved data structure that combines the advantages of the merkle tree and patricia tree. Patricia tree is a modification of the prefix tree, and the data values are stored in a single node. To improve the defects of the merkle tree and patricia tree, the waste of space controls the difficulty of nodes, long depth and etc. The following aspects have been mainly done:

- The key values are first hashed and converted to a certain length to prevent wastage due to different key lengths.
- All nodes are divided into four categories, including empty node, branch node, leaf node and expansion node, which helps to facilitate the control of nodes.
- In order to shorten the depth of the tree, the expansion nodes of more than two consecutive levels are merged.

In Ethereum, account transaction information, status, corresponding status changes, and the related transaction information are all managed by MPT, which is an important part of the entire data storage. Compared to the merkle tree, the MPT would be more effective in integrity checking of data [6].

In addition, MPT provides the Merkle Proof method to verify the integrity of the attribute set [7]. In a classical merkle hash tree, any change in the original data of the leaf nodes causes a change in the hash node. By comparing the hash values stored in the root, it is possible to know whether the data has been modified or not. However, there are three logical nodes in the MPT attribute tree [7]. The node hash is a hash calculation of a specific logical node, which is a combination of the lower hash and the ordinary merkle tree. Therefore, when performing the Merkle Proof function, the hash value needs to be computed according to the logical structure of MPT. The Merkle Proof method that MPT can provide allows for the scaling of light nodes. Light nodes are proposed to solve the problem of large data volume in Ethereum. Such nodes only need to maintain all the block header information in the chain. In the public chain environment, a light node can prove whether a transaction exists in the blockchain, whether an account exists in the blockchain, and what its balance is using the block header information maintained locally. Merkle Proof means that a light node initiates a proof request to a full node, asking whether a specified node exists in the full merkle tree of the full node. The full node returns a Merkle Proof path to the light node, which is calculated by the light node to verify the existence. The MPT tree is abstracted as shown in Figure 1, where the lowermost level is the hash of the leaf nodes, and the root hash of the tree is generated by providing the node hash at each step from the bottom up. If a light node wants to verify whether the tree node14 exists in the merkle tree, it only needs to send a request to the root node, and the heel node will return a path of node13, node21, and node32 (the merkle path, as shown in the yellow box in Figure 1). After getting the path, the light node uses node13 to hash with node14, node21, and node32 separately, and compares the result obtained with the locally maintained root hash. If the results are equal, the verification is passed.

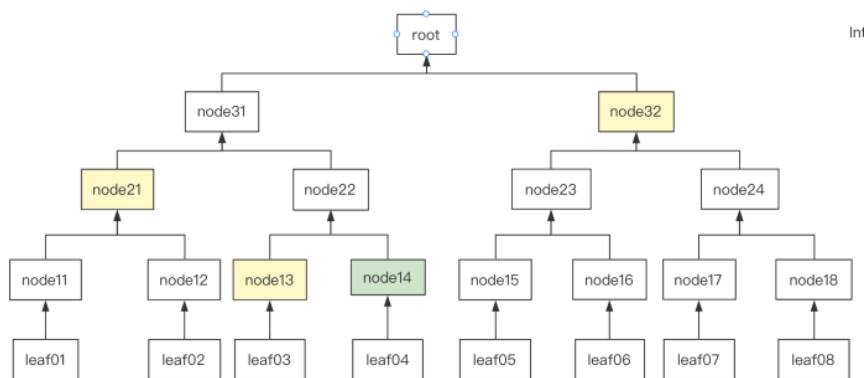


Figure 1. Merkle Tree form of MPT

Merkle Proof has strong security. When an attacker tries to forge a legitimate verification path for a node that does not exist in the blockchain, the final computation result will be the same as the merkle root hash in the block header. In this case, the unpredictability of the hash computation makes it impossible for a malicious full node to forge a path for a non-existent node, such that the final computed root hash maintains the same value. Moreover, since there is no way to determine whether the requested full node is a malicious node in the public chain environment, the result of a direct request to one or more full nodes cannot be guaranteed. However, the block header information maintained locally by the light nodes can be verified by workload proof. If the merkle path provided by the full node is used to perform hash computation with the verified node, and the final result is consistent with the root hash in the locally maintained block header, it can prove that the node exists in the merkle tree.

(3) Merkle Tree performance analysis and related improvements

Huang Gen, Zou Yibo and Xu Yun [8] have compared and verified the performance of the merkle tree in different blockchains, evaluating the performance of the merkle tree in terms of storage space, build time and SPV verification in Bitcoin, blockchain and superledger. The experiments show that in terms of storage, MPT has the highest number of nodes among the three types of merkle tree, and its depth is much higher than the other two types and is stable, not increasing with the number of nodes. In terms of build time, MPT takes the longest time to create, which is due to the specificity of its algorithm. In terms of SPV verification, MPT has a medium-length verification path, considering that SPV verification sometimes passes not only the data on the path but also other data in the hash bucket. Therefore, compared to Bitcoin, Ethereum may be relatively unsuitable for mobile use.

4. Gas And Block Size

4.1. Gas

(1) The concept of Gas

The Ethereum network cannot function without Gas. It is the fuel that permits it to function, much as a car needs gasoline to run. The unit of measurement for the amount of computing effort necessary to perform various activities on the Ethereum network is called “Gas”. Since each Ethereum transaction necessitates the use of computing resources, each transaction demands a charge. Additionally, Gas also refers to the fee that must be paid in order to complete an Ethereum transaction correctly. Figure 2 presents the mechanism for Gas generation.

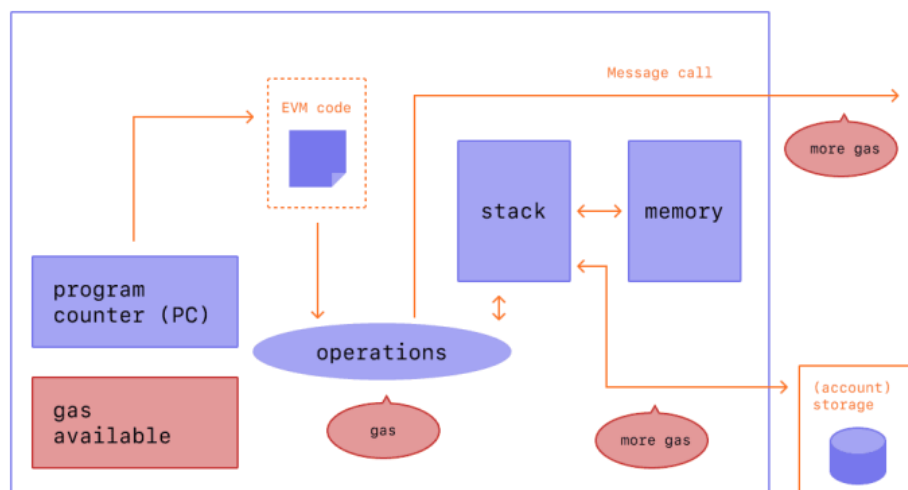


Figure 2. Mechanism for Gas generation [9]

(2) The London update

The London Upgrade in August 2021 modified the way transaction fees were calculated on the Ethereum network. By redesigning Ethereum's transaction-fee-mechanism, the London Upgrade makes transacting on Ethereum more predictable for users. This modification has a number of high-level benefits, including better transaction cost estimation, typically faster transaction inclusion, and the ability to balance ETH issuance by burning a percentage of transaction fees. Users are expected to put a gratuity in their transactions because the transaction charge's base cost has been burned.

The total transaction fee can be calculated as Formula 2 as follow [10]:

$$\text{Gasunits(limit)} \times (\text{Basefee} + \text{Tip}) \quad (2)$$

4.2. Block size

Ethereum had fixed-size blocks before to the London update. During periods of strong network demand, these blocks operate at full capacity. As a result, consumers are frequently forced to wait for a drop in high demand before being able to fit into a block, resulting in a terrible user experience.

Ethereum now has variable-sized blocks thanks to the London update. Each block has a goal size of 15 million gases, but it can grow or shrink based on network demand, up to a maximum block size of 30 million gasses (twice the target block size). The average size of equilibrium blocks is 15 million. This indicates that if the block size exceeds the desired block size, the protocol will raise the next block's basic charge. Similarly, the protocol reduces the base cost if the block size is smaller than the intended block size. The basic fee amount should be adjusted in proportion to the distance between the current block size and the target size [10].

4.3. Gas fee

(1) Base fee

As a reserve price, each block has a base fee. The price offered per gas must be at least equivalent to the base fee to qualify for inclusion in a block. Users should expect transaction fees to be more predictable because base fees are calculated independently of the current block and are defined by past blocks. This base fee is "burned" when the block is mined, removing it from circulation. The basic fee is calculated by comparing the preceding block's size (the total amount of gas consumed for all transactions) to the goal size. The base charge per block will increase by up to 12.5 percent if the desired block size is surpassed. This is an exponential cost growth, and if the block size is maintained at a sufficient amount over a lengthy period of time, the cost will become unbearable. As shown in Table 1, if the block size stays large, the cost increases exponentially.

Table 1. Basic Cost Increases for Each Block [10]

Block Number	Included Gas (M)	Fee Increase (%)	Current Base Fee (gwei)
1	15	12.5	100
2	30	12.5	100
3	30	12.5	112.5
4	30	12.5	126.6
5	30	12.5	142.4
...	30	12.5	...
30	30	12.5	2705.6
...	30	12.5	...
50	30	12.5	28531.3
...	30	12.5	...
100	30	12.5	10302608.6

Compared to the prior London Gas auction market, this modification in the transit-fee method makes cost forecasting more trustworthy. At the same time, since base fees climb at a quick rate during the course of the block, a sustained peak across the block is uncertain.

(2) Priority fee (tips)

Miners used to get the total Gas cost from any transaction included in a block before the London Upgrade. The London Upgrade provided a priority charge (tip) to motivate miners to include a transaction in the block after the new base fee was burned. Miners would find it economically advantageous to mine empty blocks if there were no tips, since they would receive the same block reward. A tiny tip gives miners with a minimal incentive to include a transaction under normal circumstances. A higher tip will be required for transactions that need to be prioritized above other transactions in the same block in order to outbid competing transactions [10].

(3) Gas Fee's role in Ethereum

One of the key advantages of the London upgrade is that it makes it easier for users to determine transaction fees. Wallet providers will automatically determine a recommended transaction fee, such as base fee and recommended priority fee, to lessen complexity put on their customers for wallets, rather than explicitly declaring how much to spend to complete transaction.

In a nutshell, charging a fee for each computation performed on it can prevent bad actors from spamming the network. Each transaction is required to specify a limit on how many computing steps of code execution it can consume in order to avoid unintentional and hostile endless loops or other computational waste in code. Gas is the fundamental unit of computation. Despite the fact that a transaction has a limit, any gas not utilized in the transaction is returned to the user (i.e. max fee - (base fee + tip)) [10]. Figure 3 presents Gas generation in the transaction.

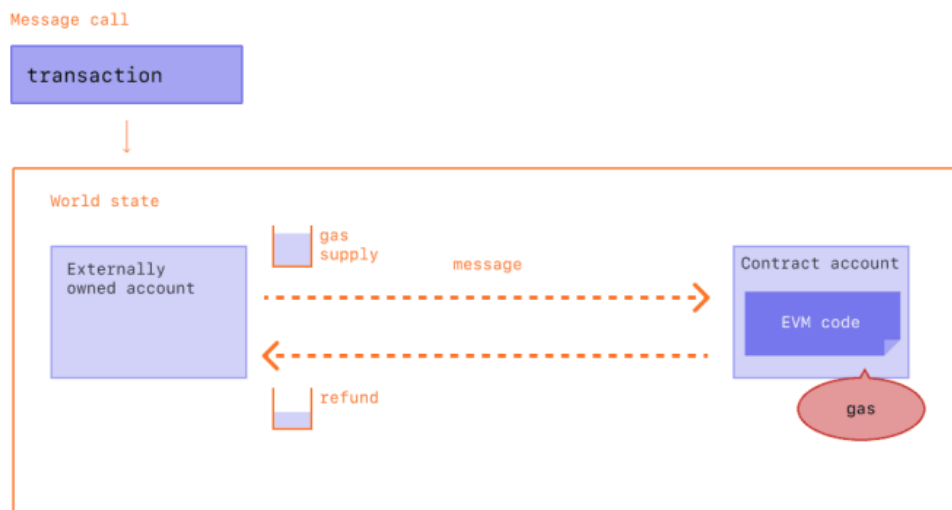


Figure 3. Gas generation in the transaction [9]

(4) The consumption of Gas fee

The high Gas fee is due to Ethereum's popularity. Any action on Ethereum necessitates the use of gas, and each block's gas space is limited. Calculating, storing, or manipulating data, transferring tokens, as well as consuming various quantities of Gas units all spend costs. The amount of operations done by smart contracts grows as dapp functions and becomes more complicated, which means that each transaction takes up more space in a block of restricted size. If there is an excessive quantity of demand, customers must pay a bigger tip amount in order to outbid other users' transactions [10]. A greater tip can increase the transaction's chances of being accepted into the following block. Using solely the gas price to decide how much to spend on a transaction is ineffective. To determine the transaction charge, the amount of gas consumed by the transaction fee is considered, which is expressed in gwei.

4.4. Gas limit

The greatest amount of Gas during a transaction is referred to as the gas limit. More sophisticated smart contract transactions necessitate more computing labor, necessitating a greater gas limit than a simple payment. A Gas limit of 21,000 units is required for a typical ETH transfer. For example, if a 50,000 gas limit is set for a basic ETH transfer, the EVM will spend 21,000 and return the rest of 29,000. However, if 20,000 Gas is set for a basic ETH transfer, the EVM will use your 20,000 Gas units in an attempt to complete the transaction. The EVM then undoes any changes, since the miner has already worked for 20k Gas units, that Gas is used [10].

4.5. Gasol for Gas analysis and optimization

Gasol is an Ethereum smart contract gas analysis and optimization tool. Gasol provides a range of cost models that is used to estimate the gas consumption associated with specific types of EVM instructions, as well as the number of times such bytecode is executed in the instruction. Only stored opcodes are used to measure the selected gas consumption in accordance with the Ethereum classification of opcodes to estimate the cost of the selected program line, and so on. Gasol returns to the user an upper bound on the cost of the function when the user selects the relevant cost model and the function of interest. Gasol employs Gas analysis to discover under-optimized storage modes and incorporates automatic optimization of chosen functions, since natural Gas consumption is typically based on instructions to access storage. Our tool works with Solidity's Eclipse plug-in, which displays Gas, command boundaries, and Gas-optimized Solidity functions [11].

5. Proof Of Work

5.1. Double-spending attack under proof-of-work

For decentralized trading platforms such as Ethereum that use the blockchain, a double-spending attack may be encountered when confirming whether the transaction is successful. The block connection method of the block chain is that after the previous block is proven, the successful miner writes the next block, so that to form the block chain. Eventually, the short chain is abandoned, and the long chain is synchronized to all nodes. This mode of operation is the theoretical mode of the proof-of-work mechanism of Ethereum. However, in actual operation, an attacker may have more than half the computing power of the entire network. In this case, a double-spend attack will occur. When a piece of money is transacted with two accounts at the same time, one of the transactions will be prioritized and written into the block under the proof-of-work mechanism, and then the next block will be opened, according to the proof mechanism of the blockchain. The verified block will logically be retained as a long chain, and any transaction that fails the verification will be abandoned. However, when the attacker has more than 50% of the computing power of the entire network, the attacker will use this huge computing power to prove the abandoned short chain. At this time, due to the huge computing power, the abandoned short chain will become a long chain and is retained as a new long chain. The original long chain was abandoned. However, since the original transaction has been

authenticated, the receiver has sold the product to the attacker. If the transaction is abandoned at this time, the recipient will lose the transaction fee that was initially verified. The attacker will get the product without spending any cost [12, 13].

5.2. Ethash against double-spending

When the Bitcoin network was first born, many mining machines designed for mining were born on the market, such as those designed for hashing algorithms. If this problem is not resolved, the network will slowly transform from a decentralized network to a centralized network. This has also changed the original intention of the virtual currency trading network. In order to deal with these mining machines specifically for mining, Ethereum improved the POW algorithm, from the original Dagger-Hashimoto to Ethash [14].

Its calculation process is to first calculate a seed by scanning a given block header, and then use this seed to generate a 16MB cache. Through this cache, a data set DAG with an initial size of 1GB can be generated. An entry in the data set can be generated by only one of the entries in the cache. The miner finally randomly selects some items in the data set to hash it together with the block header and nonce. Seed is actually a hash value. It is obtained by superimposing the Keccak hash multiple times [14].

Each Epoch has 30,000 blocks. The hash value of the seed in the first Epoch is empty, and the hash value of the seed in each subsequent period is obtained by hashing the seed hash value of the previous period through Keccak256. Since the seed hash value is updated every cycle, the data set of each Epoch cycle will also be updated accordingly. After getting the seed hash value of the second Epoch, the seed hash value keccak512 is hashed, and this result is written into the first group. Each subsequent group of data passes through the previous group of data. The keccak512 hash calculation is derived. It merges all the arrays together and executes the RandMemohash algorithm for 3 times. After getting the cache, it can generate a data set. After concatenating the hash value of the new block header and the nonce, it performs a KEC512 hash to obtain a seed, and then initializes a 128-byte mix array. Each value in the mix array comes from the seed. The specified fragments from the data set can be read and written randomly. This operation needs to be repeated 64 times, and then it is overwritten on the mix and the new mix is compressed to obtain the hash value of the mix. After splicing the seed and mixhash, it can perform KEC256hash to obtain a random number N [14].

5.3. Difficulty changes

The difficulty value is a value calculated from the parent block header and the new block header. Its function is to control the stability of the mining rate. The difficulty value will dynamically change according to the verification speed of the previous block. For example, if the mining speed of the first few blocks is too fast, the difficulty value required for verification in the next mining will increase. The subsequent mining speed will be reduced. In simple terms, the mining difficulty value controls the difficulty by controlling the verification value range of the hash value obtained by the verification mining. If the difficulty needs to be increased, the verification value range will be reduced. However, as the difficulty value changes dynamically, a problem also arises. More people want to become miners for Ethereum. The influx of a large number of miners speeds up block verification. In order to control the mining speed, it is necessary to increase the difficulty value. As the difficulty of mining increases, miners will purchase a large number of professional mining machines in order to get the first gain from verification among the many miners. As the difficulty value continues to rise, the computing power of ordinary home computers can no longer support the miners. The dominant power of mining will be professional miners with professional mining machines instead of the original Ethereum envisioned. Everyone can join Ethereum, and everyone can become a decentralized miner. When the computing power is in the hands of a small number of people, this platform will slowly become a centralized platform, and those who have the computing power may become attackers to implement attacks such as double-spend attacks [15].

6. Conclusion

In order to manage account transaction information, status, and the corresponding state changes and to improve data security, Ethernet has introduced MPT to store the relevant data information. MPT is more effective than the traditional Merkle Tree in checking the integrity of data. However, previous experiments have shown that MPT is relatively unsuitable for mobile devices due to its high number of nodes and long creation time. Merkle Proof method can be provided to achieve the extension of light nodes, thereby supporting simple payment verification. Gas component is a function of fuel. Gas is the unit of measurement for the computational effort necessary to carry out various tasks on the Ethereum network. Each Ethereum transaction is taxed, since it consumes computational resources. By charging for each activity performed, one of the most significant responsibilities of Gas is to maintain the security of Ethereum's computational power and avoid many redundant dangerous operations. In this paper, the method of calculating the cost of Gas is discussed in detail. Furthermore, this paper analyzes how block size can adjusted to avoid waste of computer power and some dangerous operations. This increases the incentive for Ethereum to expand. Facts have proved that the proof-of-work mechanism only increases the cost of verifying blocks, so that attackers need to pay a higher price to carry out double-spending attacks. As long as the Ethereum network continues to use the proof-of-work mechanism on the blockchain platform to verify blocks, it cannot effectively avoid 51 attacks. This paper discusses the reasons why Ethereum plans to change from the pow to the pos mechanism. The dynamic change of the difficulty value also makes the mining cost increasingly higher, and it is difficult for home computers to have the mining function, which also makes the Ethereum network tend to be centralized. Therefore, the Ethereum network platform still has many security risks that need to be improved.

References

- [1] Miller, V. S. (1985, August). Use of elliptic curves in cryptography. In Conference on the theory and application of cryptographic techniques (pp. 417-426). Springer, Berlin, Heidelberg.
- [2] Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*, 48(177), 203-209.
- [3] Sun Songyan, Cai Changshu, Zhou zhou, Zhao yanjie, & Yang Jinming. (2013). Comparative Analysis and Research on National Secret SM2 Digital Signature Algorithm and ECDSA Algorithm. *Network Security Technology and Application*, (2), 60-62.
- [4] Yong, Y., & Wang, F. Y. (2016). Security Problems on Blockchain: The State of the Art and Future Trends. *Zidonghua Xuebao/Acta Automatica Sinica*, 42(4), 481-494.
- [5] Dinh, T. T. A., Liu, R., Zhang, M., Chen, G., Ooi, B. C., & Wang, J. (2018). Untangling blockchain: A data processing view of blockchain systems. *IEEE transactions on knowledge and data engineering*, 30(7), 1366-1385.
- [6] Mathiyalahan, S., Manivannan, S., Nagasundaram, M., & Ezhilarasie, R. (2018). Data integrity verification using MPT (Merkle patricia Tree) in cloud computing. *Int. J. Eng. Technol.*, 7(2.24), 500-503.
- [7] Zheng Xin, Huang Dexiao, & Wang Gaocai. (2021). Attribute access control strategy based on mpt modeling. *Computer Application Research*, 38(4), 8.
- [8] Huang Gen, Zou Yibo, Xu Yun. (2020). A study of merkle tree performance in blockchain. *Computer Systems Applications*, 29(9), 7.
- [9] Takenobu, T. (2018, March 10). Ethereum EVM illustrated. https://takenobu-hs.github.io/downloads/ethereum_evm_illustrated.pdf.
- [10] Douglas, J. (2021, December 23). Gas and fees. <https://ethereum.org/en/developers/docs/gas/>.
- [11] Albert, E., Correias, J., Gordillo, P., Román-Díez, G., & Rubio, A. (2020). Gasol: Gas Analysis and Optimization for ethereum smart contracts. *Tools and Algorithms for the Construction and Analysis of Systems*, 118–125. https://doi.org/10.1007/978-3-030-45237-7_7.
- [12] QI. (2020, April 23). Proof-of-work in Ethereum. <https://learnblockchain.cn/article/906>.

- [13] PANews. (2021, December 8). Ethereum double spending. <https://www.panewslab.com/zh/index.html>.
- [14] Blockchain Technology Apprentice. (2019, October 31). What is 51% attack. <https://www.cnblogs.com/blockchainnote/p/11769356.html>.
- [15] Raynor de Best. (2021, December 7). Ethereum (ETH) mining difficulty up until December 7, 2021. <https://www.statista.com/statistics/1224536/mining-difficulty-ethereum/>.