DMO, an anti-quantum PoT distributed ledger

Abstract

The rapid development of quantum computer and its threat to classical cryptography are the inevitable factors for the research and development of DMO as a new distributed ledger with anti-quantum properties. Generally speaking, the block chain technologies such as DMO and BTC, which adopt elliptic curve signature algorithm, are completely different in the signature layer because they adopt anti-quantum algorithm. In the face of quantum threat, elliptic curve algorithm will be eliminated, and all blockchain systems need to open new routes in the form of bifurcating or reconstructed mapping, and the DMO is the initiator of this great era of navigation. Since all anti-quantum signature algorithms are larger in size than elliptic curve signature algorithms, the DMO also adopts a new consensus mechanism based on proof of Trust, which we call PoT (Power of Trust).

Background

The latest progress in the field of quantum computing, and its threat to classical cryptography, especially chain block areas may be one of the most affected areas, because of a quantum computer can gain economic incentives, block chain accounts can be anonymous access and BTC, for example a large part of the data will be cracking moment, quantum damage and even cause the entire block chain system crisis of confidence, thus promote the us against quantum research interest.

Quantum computing is a complex concept, but in a nutshell, it is a way of encoding and associating information on a large scale.The realization of quantum computing will change the world communication, finance and management and protection of data-this is a radical change of the network security situation in technology development, construct cryptosystem is resistant to quantum attacks after the quantum cryptography system, provide protection scheme for the network information security and block chain, public key cryptography is an important research direction. This paper deals with the block chain technology, especially the encryption method and system of digital signature of block chain against quantum computing attack. The method includes but is not limited to the following:

S1: Key parameters used to generate digital signatures;

S2: When the transaction is initiated, the transaction data is digitally signed;

S3: Verify the digital signature and receive the transaction data;

The purpose of this study is to provide an encryption method and system for digital signature of blockchain against quantum computing attacks, so as to solve the authenticity and non-repudiation of transaction data sent in the blockchain system in the existing technology, and solve the problems such as the inability of the existing blockchain algorithm to resist quantum attacks.

Catalog

1 Development status of quantum computing

Quantum Computing takes Quantum bit as the basic unit and realizes data storage computation through the controlled evolution of Quantum state, which has incomparable information carrying capacity and parallel processing capacity compared with classical Computing.

Quantum computers work on the basis of how subatomic particles behave as described in quantum mechanics. Quantum mechanics is the branch of physics that deals with subatomic particles such as electrons.

1.1 Quantum information carrying capacity

Due to the existence of quantum superposition states, quantum bits can carry more information than classical bits.

Quantum superposition: A quantum system can be in a superposition of different quantum states without interference from outside observations, known as Schrodinger's cat.

Qubit: A quantum bit can represent a superposition of 0, 1, or 0 and 1, so it carries far more information than a classical bit that can only represent 0 or 1.

An N classical bit store can store only one of 2^n possible n bit numbers, while an N qubit store can store 2^n numbers at the same time, and its ability to store information increases exponentially with n.

1.2 Quantum parallel processing capability

Because of the quantum superposition effect, the unitary transformation in quantum computation can operate simultaneously on all components in the superposition state. Therefore, quantum computers can simultaneously multipath parallel operations, which is also the source of quantum computers' super information processing power.

In classical computation, the so-called parallelism is only a time-sharing processing of a certain state.

In quantum computation, because of the superposition state property of quantum bits, all possible states can be processed simultaneously.

1.3 Advantages of quantum computing

Quantum computing is indeed particularly good at two kinds of computing problems:

One type of unstructured search. Grover algorithm in the field of quantum computing can directly square the complex impurity of unstructured search linear algorithms and realize exponential acceleration compared with classical computation. Most importantly, the algorithm halves the strength of protection against scale keys and hashes, weakening the security of encryption algorithms such as AES-256, AES-128, SHA2-256, and Sha2-128. To resist quantum computing attacks, symmetric ciphers and hashes have to double their key lengths. Another class of problems where quantum computers have great advantages is the basis of modern public-key asymmetric algorithms for solving equations that use large prime numbers in mathematical formulations. In 1994, quantum physicist Peter Shor showed that such problems could be solved with a quantum computer in seconds to minutes. With enough qubits, today's asymmetric secret codes and digital signature schemes will collapse.

The first working quantum computer appeared in 1998 with just two qubits. But even with these two qubits, the quantum computing team has shown that Shor's algorithm can solve equations of any size as long as it has enough stable qubits. Since then, all quantum computer makers (more than 100 different teams are working on quantum computers) have been trying to add qubits and stability (error-correcting capability) to their machines. For years, quantum researchers have publicly announced any increase in the size (or stability) of qubits. But starting in 2018, they stopped updating the news of incremental growth and declared themselves closer to quantum power.

Quantum computer can finally traditional binary classical computers can't -- unprecedented computing speed, and solve mathematical problems that conventional computers cannot, if today the quantum computer has reached quantum hegemony just across a layer of the veil, that is to say, HTTPS, TLS, WiFi, digital certificate, smart CARDS, FIDO authentication token or is about to be cracked, encryption, currency, etc., or could have been cracked. The world's most capable countries are pouring money into quantum technology research and development, and the age of quantum computing will soon begin. In either case, finance, blockchain, or every internet-related industry needs to start thinking hard about its own post-quantum security and start preparing for tomorrow's threats today.

2 The impact of quantum computing on cryptography

Quantum computers based on Shor algorithms are more effective than conventional computers at breaking down encryption based on Rives-ShaMir-Adleman (RSA) and elliptic curve cryptography (ECC). These two types of encryption are the most common ones, and they are also the security cornerstones of anti-quantum encryption urgently needed.

Digital signature is one of the basic elements of public key cryptosystem. Since integer decomposition and discrete logarithm problems have been proved to be highly effective in quantum computer, the traditional public key cryptosystem based on the above difficult problems is not secure in quantum computer, such as information security and blockchain.

2.1 Influence on password system

The classic problem in security solutions is encrypting, decrypting, signing and validating transactions and data. Using quantum computers, attackers have higher processing power and quantum-based algorithms like Shors to crack cryptographic systems.

2.1.1 Influence of symmetric cryptography algorithm

1) Grover algorithm

Grover random database search algorithm in quantum computation can improve the efficiency of spatial search by "exponential halving". If the security of symmetric cipher is measured by the attack complexity of exhaustive search key, the security of symmetric cipher will be reduced by half, that is, symmetric cipher algorithm that is 2^128 secure in classical computing environment is only 2^64 secure in quantum computing environment.

Impact: Quantum computing can reduce the security of DES, AES, SM4 and other block cipher algorithms and stream cipher algorithms to 1/2 of the original. However, because Grover algorithm requires exponential memory, it has little effect on symmetric cipher algorithm.

2) Coping strategies

Increase key length: In order to achieve the quantum computation security of 2^128, the effective cipher size of symmetric cipher should be designed to be at least 256 bits. It should be noted that the increase of key length has an influence on the speed of encryption and decryption operation, but the influence can be controlled, from about 450MB/s of 128bit key length to about 320MB/s of 256bit key length.
AES algorithm: support 256 bit key length, can temporarily meet the security of quantum computing.
SM4 encryption algorithm: the key length is fixed at 128 bits, which may be affected.

2.1.2 Influence on hashing abstract algorithm

1) Quantum random walk algorithm

The quantum random walking algorithm can improve the time efficiency of the classical search algorithm, and then increase the probability of finding two original collision images randomly within a certain time, so as to realize the violent crack against collision and reduce the security of hashing abstract algorithm.
Impact: Quantum computing reduces the security of hash summary algorithms such as MD5, SHA, SM3 to 2/3 of the original level.

2) Coping strategies

Increase the abstract length: Quantum computation has little influence on the hash abstract algorithm, so only the algorithm with large abstract length can be used.
SHA algorithm: the highest support SHA-512, can temporarily meet the security of quantum computing. Sha-256 is currently considered safe in canonical computing environments, so at least SHA-384 should be used in quantum computing environments.

SM3 encryption algorithm: The length is fixed to 256 bits, which may be affected.

2.1.3 Influence on public key cryptography

1) Shor algorithm

Shor quantum algorithm can solve large integer decomposition problem and discrete logarithm problem in polynomial time. Whereas a classical computer would take more than a billion years to crack the 2048-bit RSA key, a Google quantum computer, which claims to have 20 million qubits, can crack the 2048-bit RSA algorithm in just eight hours.

Influence: Quantum computing can crack public key cryptography algorithms based on large integer decomposition such as RSA and ECDSA and SM2 based on discrete logarithms such as ECC elliptic curve public key cryptography algorithms.

2) Coping strategies

Replace the new algorithm: the difficult problems of large integer decomposition and discrete logarithm can be easily solved in the quantum parallel computing environment. Therefore, it can be considered to select the mathematical difficult problems that cannot be effectively parallel computing to construct anti-quantum public key cryptography algorithm.

2.2 Impact on blockchain and Distributed Ledger Technology (DLT)

Traditional blockchains, such as Bitcoin and Ethereum, use classic public-key cryptography to sign transactions, and these networks are thought to be vulnerable to quantum computing attacks. Other systems, such as zCash and Quorum, rely heavily on special elliptic curves to provide zero-knowledge proof, and an ECC breach threatens the integrity of these books.

This would be a major security concern, and it would lead to a complete breach of the network.

2.3 Anti-quantum cryptography

2.3.1 Definition of anti-quantum cryptography

Anti-quantum Cryptography, also known as post-quantum Cryptography, is a Cryptography system that can resist both existing classical computing attacks and future Quantum computing attacks. It should at least meet the following requirements:

Able to resist known classical attack methods;

Can resist known quantum computing attack methods (such as Shor algorithm);

There is no known quantum attack method that successfully attacks a cryptographic algorithm in a polynomial.

Some difficult problems cannot be solved by quantum discrete Fourier transform, such as SVP/CVP on lattice, solution of nonlinear equations, knapsack problem, etc. Therefore, anti-quantum cryptography algorithm can be constructed based on such problems.

Anti-quantum cryptography algorithm should meet the following five conditions:

（1） Security: Not only in today's computing conditions, but also in the quantum computer security.

(2) Fast running speed: The existing results show that the computing speed of post-quantum cryptography can exceed that of the existing public key cryptography under the same security intensity.

(3) Reasonable communication overhead: in practice, an algorithm with a public key/ciphertext size of several M or larger will hardly be used. The current RSA-2048 public key size is about 256Bytes, and the elliptic curve is about 64Bytes. The public key size of the best post-quantum cryptography algorithm is around 1KB.

(4) It can be used as a direct substitute for existing algorithms and protocols, such as public key encryption, key exchange, digital signature, etc.

(5) More scenarios: For example: Homomorphic Encryption, attributive based Encryption, Functional Encryption, Indistinguishability Obfuscation, and other advanced cryptography applications.

2.3.2 Research status of anti-quantum cryptography

At present, most of the cryptosystems attacked by quantum computers are the first generation of public key cryptography, including RSA / ECC / DH mentioned above. And these public key cryptography just constitute the anchor of trust chain in contemporary cyberspace. Therefore, the focus of people's attention at this stage is to come up with a solution that can replace the first generation of public key cryptography as soon as possible, so as to re fix the anchor of trust in cyberspace.

In the international cryptography community, four types of mathematical ciphers, namely hash based, coding based, multivariable based and lattice based, are collectively referred to as "post quantum cryptography" (now "anti quantum cryptography"). There are two main reasons why they are given the title of "anti quantum"

1、 The mathematical difficulties they rely on are not related to the difficult problems solved by shor algorithm, which are relied on by the first generation public key cryptography algorithm. In other words, shor algorithm has no effect on them;

2、 In the future, if one of them can be solved, other algorithms can still be used as a supplement.

Since there are so many different forms of anti quantum algorithms, why don't we replace the existing public key cryptosystem with them immediately?

Although these anti quantum cryptography algorithms rely on different mathematical difficulties and thus have the characteristics of resisting quantum computer attacks in theory, compared with the first generation of public key cryptography algorithms, they have some defects, such as low efficiency, large key size, slow encryption and decryption speed, etc. Once they are rashly put into today's Internet, it may bring about a significant decline in operating efficiency. There is no doubt that people will not want to use such an encryption algorithm, which will take hours to verify whether Microsoft's official website is credible before deciding whether to download the latest patch. Another drawback is that none of these algorithms can integrate encryption, signature and authentication into one. This is exactly one of the advantages of the previous generation of public key cryptography algorithms. More importantly, the ultimate goal of any cryptographic algorithm is application, and the application should be oriented to different applications in all aspects of the Internet. Anti quantum cryptography algorithm is bound to face more, newer and more complex network applications, including mobile Internet, satellite communication, Internet of things, big data, cloud computing and so on. For example, are various "lightweight anti quantum cryptography" for Internet of things terminals enough to resist "universal quantum computer attacks" and have fast encryption and decryption speed? For example, how can these anti quantum cryptography smoothly transition to the existing network security protocol stack (including TLS, Ike, etc.) without affecting the operation efficiency of the network?

In recent years, the importance of post quantum cryptography has become increasingly urgent. Taking the United States as an example, NIST PQC standard collection work was officially launched in 2016. NIST focuses on the following three types of post Quantum Cryptography: encryption, key exchange and digital signature. It has been two years since NIST published it. The cryptography community has carried out detailed cryptanalysis on these algorithms. At present, nearly 1 / 3 of the algorithms have been found to have various defects, and nearly 1 / 5 of the algorithms have been completely broken.

At present, there are mainly four ways to research post quantum cryptography algorithm

(1) Hash-based: It first appeared in 1979 and is mainly used to construct digital signatures. Representative algorithms: Merkle hash tree signature, XMSS, Lamport signature, etc.

(2) Code-based: It first appeared in 1978 and was mainly used to construct encryption algorithms. Representative algorithm: McEliece.

(3) Multivariate based: It first appeared in 1988 and was mainly used to construct digital signature, encryption, key exchange, etc. Represent methods/algorithms: HFE(Hidden Field Equations), Rainbow (cooled Oil and Vinegar (UOV) methods, HFEV, etc. (4) Lattice based: It first appeared in 1996. It is mainly used to construct encryption, digital signature, key exchange, and a lot of advanced cryptography applications, such as attribute-based encryption, Trapdoor functions, Pseudorandom functions, Homomorphic encryption, etc. Representative algorithms: NTRU series, NewHope (tested by Google), a series of homomorphic encryption algorithms (BGV, GSW, FV, etc.). It is considered as the most promising post-quantum cryptography technology because of its fast computing speed, low communication cost, and its ability to be used to construct various cryptography algorithms and applications.

When the parameters are selected properly, there are no known classical and quantum algorithms to solve these problems quickly.

The security of these algorithms depends on whether there is an efficient attack algorithm that can quickly solve the underlying mathematical problem or directly attack the algorithm itself. This is why quantum computers pose a great threat to public key cryptography.

In addition to the four kinds of ways, and based on very singular elliptic curve (Supersingular elliptic curve isogeny), Quantum random walk (Quantum walk) technologies such as Quantum cryptography after construction method, the most important thing is based on the hash, based on the coding, based on the multivariable, based on the grid, such as four, because as mentioned earlier, the Quantum cryptography mainly focus on the public key cryptography. These four approaches are the best way to construct post-quantum versions of all kinds of algorithms in public key cryptography, and even beyond (such as lattice-based (full) homomorphic encryption).

2.3.3 Main anti-quantum cryptography technologies

Anti-quantum cryptography is the study of cryptography system. The DMO project team studied the process of NIST initiating the standardized quantum encryption, and carried out technical research and comparison on the most promising cryptography system including lattice-based, multivariable, hashing and coding.

Before diving into the various cryptographic technologies, we briefly summarize the trade-offs inherent in each type of cryptographic system and compare them with current (non-post-quantum) elliptic curve cryptography.

2.3.3.1 Based on Lattice

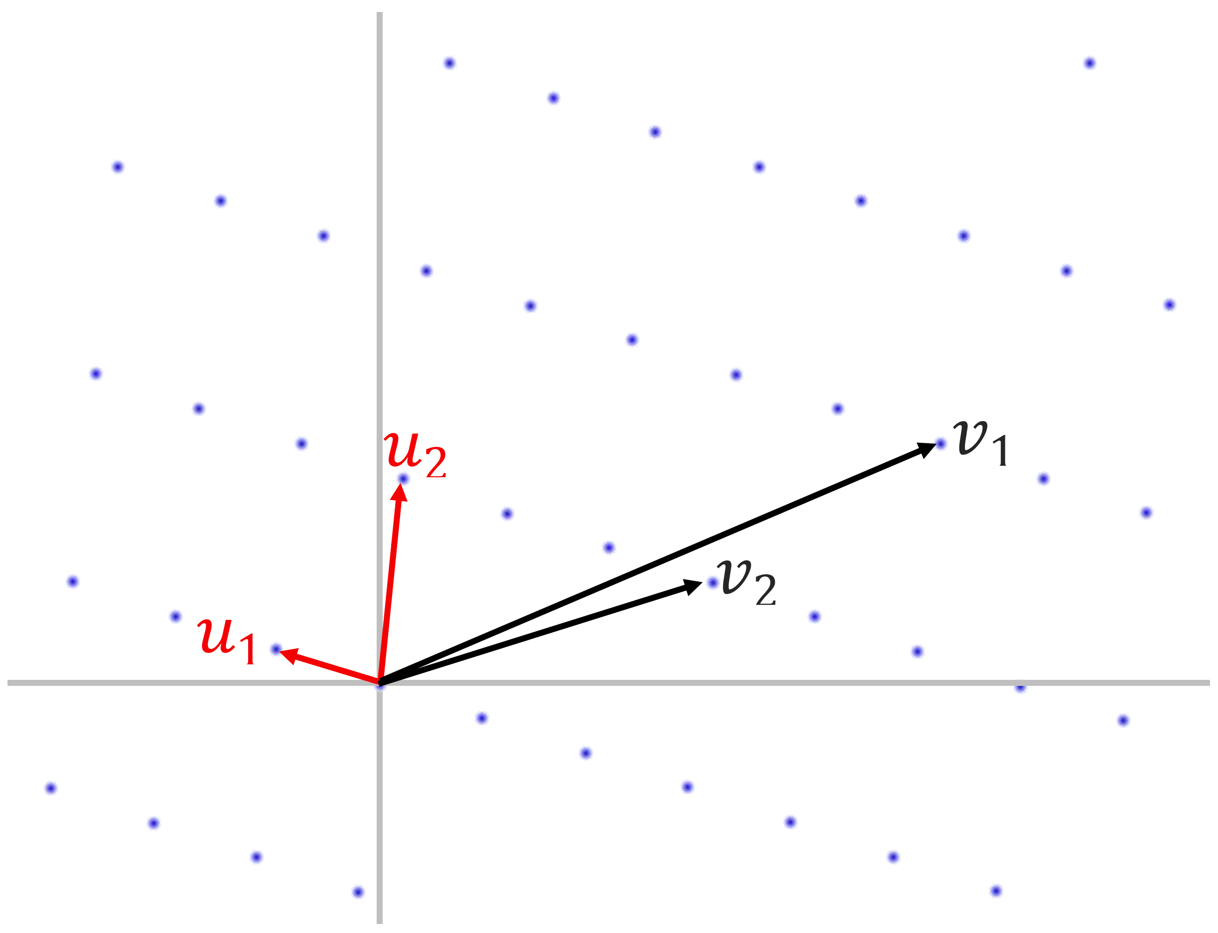
Lattice Based Cryptography is the most representative type of anti-quantum public key Cryptography. Lattice is a kind of algebraic structure parallel with groups, rings and fields, and LWE is a difficult problem in lattice. In 2005, Regev proposed the LWE problem and reduced THE LWE to the shortest vector difficulty problem (SVP) on the lattice by quantum algorithm, and gave the public key cryptography scheme based on LWE. LWE is more convenient to build a cryptographic system than the lattice difficulties that have appeared before.

Ring-lwe is a variant of LWE. In ring-LWE-based cryptography, the key is represented by several polynomials instead of the matrix in LWE, which reduces the size of the key and increases the encryption and decryption speed.

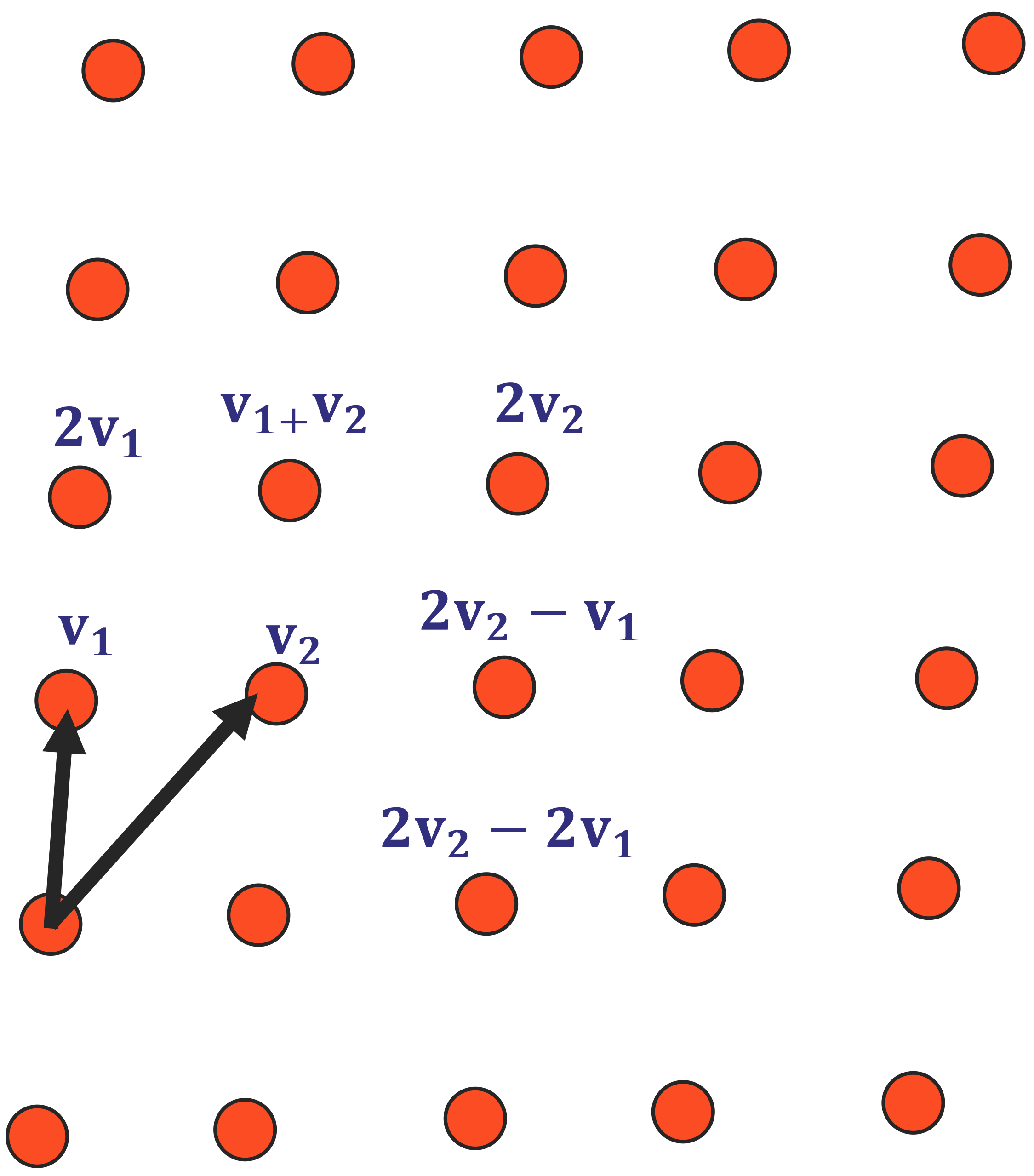
The lattice-based algorithm is considered as one of the most promising post-quantum cryptography algorithms due to its better balance in security, public-private key size and computing speed. Compared with the construction of cryptographic algorithms based on number theory, lattice-based algorithms can achieve significantly improved computing speed, higher security intensity and slightly increased communication overhead. Compared with other post-quantum cryptography methods, lattice-cipher has smaller public and private key sizes and better indicators such as security and computing speed. In addition, the lattice-based algorithm can realize encryption, digital signature, key exchange, attribute encryption, function encryption, homomorphic encryption and other existing cryptographic structures. The security of lattice-based algorithms depnds on the difficulty of solving lattice problems. At the same (or even higher) level of security, the lattice-based algorithm has smaller public-private key sizes than the three constructs described above, computes faster, and can be used to construct multiple cryptographic primitives, making it more suitable for real-world applications.

In recent years, the structure of lattice cryptography based on LWE(Learning with Errors) and RLWE (Ring-Lwe) has developed rapidly and is considered as one of the most promising technology routes to be standardized.

Lattice is a mathematical structure, defined as a linear combination of integral coefficients of linearly independent non-zero vectors (called lattice basis). Specifically, given a lattice basis, for any integer, that is a vector that is a member of that lattice, n is called the dimension of the lattice. For example, the figure below shows a two-dimensional lattice and two different sets of lattice bases:



A lattice is a collection of discrete points in an N-dimensional linear space, each element of which is a vector. In n-dimensional linear space, m(m ≤ n) linearly independent vectors (B1, B2... The set of vectors generated by bm is called lattice. Vector group (b1, b2,... Bm) is a group of bases of the lattice; The number of vectors in the basis of a lattice is called the dimension of the lattice; You can have multiple different bases for the same cell, but the basis has the same dimension.



The closed lattice cipher is based on V1 and v2, and all the points in space can be represented linearly by V1 and v2.   Points in real space are infinite.

A lattice grid foundation may not be the only, for example, ((2, 1), (1, 1)) and ((1, 0), (0, 1)) is a set of two dimensional integer frames. As can be seen from the figure above, even two sets of lattice bases that define the same lattice can vary greatly in length. Mathematicians and cryptographers generally believe that for a lattice of sufficiently high dimension, it is difficult to find a set of short lattice bases or to obtain a set of linearly independent short lattice vectors through a set of randomly selected lattice bases. This problem is called the shortest independent vector problem (SIVP). In addition, there are other difficult lattice - based problems, such as Gap-SVP, BDD.

Of all the methods of post-quantum encryption, the study of lattice is the most active and flexible. They have strong security and can be used for key exchange, digital signature, complete homomorphic encryption and other more complex constructs. Although the optimization and security proof of a lattice cipher system require extremely complex mathematics, the basic idea is that it requires only basic linear algebra. Suppose you have a system of linear equations of form, such as:

Solving x is a classical linear algebra problem, which can be solved quickly by gaussian elimination

.

$$f_x(a) = a_0x_0 +\cdot\ + a_nx_n$$

Given a vector a, we see the result ax, but we don't know x. After querying this function enough times, we can get f in a very short time (by solving the above system of equations). In this way we can redefine the linear algebra problem as a machine learning problem.

Now, suppose we introduce a little bit of noise to our function, so that when we multiply x and A, we add an error term e, and reduce the magnitude of the whole thing by one (medium) q.

$$f_x(a) = a_0x_0 +\cdot\ + a_nx_n + \epsilon \quad mod\ q$$

Before studying how to use LWE for anti-quantum cryptography, we should point out that LWE itself is not NP-hard. It doesn't go straight down to SVP, but it goes down to an approximation of SVP, presumably whether it's NP-hard. However, there is no polynomial (or subexponential) algorithm for solving LWE.

Now let's use the LWE problem to create an actual cryptographic system. The simplest solution, created by Oded Regev in his original paper, demonstrates the hardness of the LWE problem. Here, the secret key is an N-dimensional vector Q with an integer entry mod, the LWE secret mentioned above. The public key is the matrix discussed earlier in A, and the output vector of the LWE function:

$$pk + \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0n} & y_0 \\ a_{10} & a_{11} & \cdots & a_{1n} & y_1 \\ & & \vdots & & \\ a_{n0} & a_{n1} & \dots & a_{nn} & y_n \end{bmatrix}$$

$$sk = \left(s_{0,}s_1,\dots,s_n\right)$$

An important property of this public key is that when multiplied by the vector, we get the approximate error term (-sk,1)0.

To encrypt some information m, we take the sum of the random column A and the code in the last coordinate of the result, m plus 0, if m is 0, q over 2 if m is 1. In other words, let's take a random vector of x0 or 1 and do the calculation.

$$Ax + (0,\mu) = c \qquad \mu = m\left\lfloor \frac{q}{2} \right\rceil$$

We evaluated the LWE function, knew that it was hard to crack, and in the output of this function proved that our bit decryption was valid because we knew that the LWE key would allow the recipient to withdraw the message, with an argument:

$$c \cdot (-sk,1) \approx A \cdot x \cdot (-sk,1) + m\left\lfloor \frac{q}{2} \right\rceil \approx m\left\lfloor \frac{q}{2} \right\rceil$$

When the wrong distribution is correctly selected, it never distorts the message beyond Q /4. The receiver can test whether the output is closer to 0 or accordingly q/2 mod Q decoded the bit.

A difficult problem

The main mathematical basis of lattice cipher is two difficult problems in lattice:

(1) The shortest vector problem of lattice (SVP): For a given set of bases, find the non-zero vector with the smallest Grenadine distance (the distance between two points) generated by it. That is, find a non-zero vector v on the lattice, satisfy any non-zero vector u on the lattice, |, |, v|, |≤|, |, u|, |.

(2) Nearest vector problem of lattice (CVP): for a given lattice and any vector, find out the vector closest to this vector in the lattice, that is, find a vector v on the lattice, satisfy any non-zero vector u on the lattice, all have ||v-y||≤||u-y||.

The difficult problem of lattices is easy to solve algeologically, but difficult geometrically.

The advantages of lattice cipher

Lattice based cryptosystem is one of the cryptosystem candidates in the post-quantum computing era.

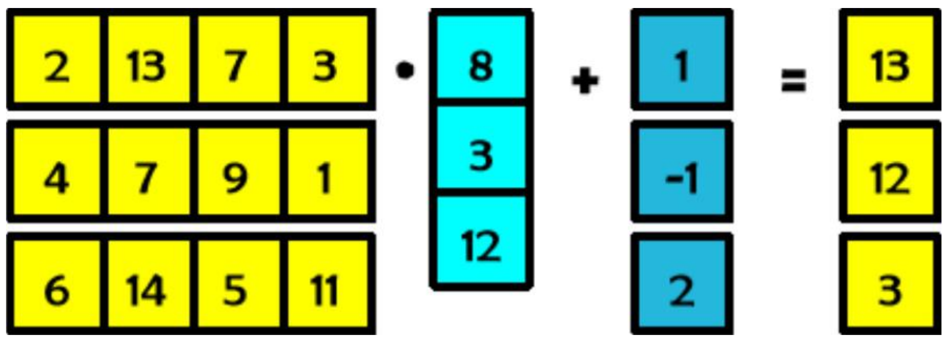
1) Because SVP and CVP of lattices are proved to be NP-hard problems under random reduction. That is, the average difficulty of some problems in a certain lattice is equivalent to the difficulty of a class of NP problems on a lattice. Therefore, the security of any instance of public key cryptosystem constructed on the basis of these lattice problems is the same as that of the most difficult instance.

2) Since lattice is a linear structure and most operations on lattice are linear, the new public key cryptosystem constructed by lattice puzzle is faster than existing schemes.

3) At present, there is no polynomial quantum algorithm to solve some lattice difficulties, so the new public key cryptosystem designed based on lattice difficulties can resist quantum attacks.

LWE algorithm

The LWE algorithm is just as hard as the worst-case lattice problem, which is considered exponentially hard (even in the



face of a quantum computer).

As+e=b; As+e= B;

As shown in the figure, A and B are known, E is A small number (equivalent to noise), and S is the key.

If there is no random noise e, it is a simple linear problem, it is easy to find the secret key, but with e added, the problem becomes NP problem. LWE problem (Learning With Errors) is a difficult problem of average sex on lattice, which can be reduced to SVP of lattice and other difficult problems, and is also anti-quantum. At present, the mainstream lattice encryption schemes are built on LWE.

The basic idea of LWE algorithm

- Parameters:
  - Q(modulus),n(dimension)m > n(# of samples)
- Secret: uniformly random vector s ∈ Z_q^m
- Input: random matrix A $\in Z_q^{mxn}, vector\ b\ \in Z_q^m$
  - E chosen from some distribution s.t. |e| $\ll q$ whp
  - B is close to the columns space of A
- Gosl: discover s

1) Take a random vector as secret key S;

2) Matrix A, vector B is known, e is A very small distribution and the modulus of E is far less than Q;

3) Calculate the value of S by b=As+ E %q.

The way to calculate s

Np-hard problem (NP) is a non-deterministic polynomial (NP). Nondeterministic means that a certain number of operations can be used to solve problems that are solvable in polynomial time. : Any NP problem can be reduced to this problem in polynomial time, but the problem itself is not necessarily NP problem. Reduction means that in order to solve problem A, we first reduce problem A to another problem B. Solving problem B also indirectly solves problem A.

Easy direction: if we can solve uniform-secret LWE

then we can solve uniform-secret LWE

- We are given A and b = As + e

  - S is a small secret

- Choose a uniform random r
- Setb’ = Ar + b = A(r + s) + e (mod q).s’ =r+s is

  uniform (because r is uniform)
- A, b’ = As’ + e is instance of uniform-secret LWE
- Solving it, we get’ s and can compute s = s’ – r

1) Now given A b and A known, the purpose is to find S;

2) Select a certain value r, such that B '=Ar+b(all parameters in this formula are known);

3) Substitute B =As+e into the above equation;

4) B '=A(r+s)+ E (mod Q);

5) - > b = 's' = r + s As + e.

6) If you solve for S ', you solve for S.

LWE encryption process

Secret key: vector s’

Public key: Matrix A’ , vector b = A’ s’ + e

- Denote Λ = (b | Λ')
- If decision-LWE is hard then A is pseudorandom
- Denote s = (1, - s’), then Λs = b - Λ’ s’ = e

Encryp$\boldsymbol{t}_{\boldsymbol{A}}(\sigma \in \{0,1\})$

- Choose a random small vector r ∈ $\{0,1\}^m$
- Output the ciphertext c = rΛ $+ \frac{q}{2} \cdot \frac{(\sigma, 0, \cdots, 0)}{\mathfrak{I}} \in$ Z_u^n

Decryp$\boldsymbol{t}_s(c)$

- Compute the inner product = (c, s)(mod q)
- Output 0 if |y| < q/4, else output 1

Example: A case process of proof of statistical zero knowledge based on grid

Prover

$r_A, r_e \overset{s}{\leftarrow} \mathfrak{y}\sigma(n \propto)$

$\iota = (lr_s) + r_e$

$(c_{anx}d_{anx})$=a Commit(l) $\xrightarrow{c_{au\lambda}}$

$\xleftarrow{c}$ c $\overset{s}{\leftarrow}$ C $= \{0,\cdots,2_n - 1\}$

$S_3 = r_s + X^C s$

$S_c = r_c + X^C c$

Accept with probability

Protocol 3.2: proot’ of knowledge of LWE-secrets s, c such that y-as + e

$as_s + s_e = a(r_s + X^c s) + (r_e + X^c e) = X^c(as + e) + (ar_s + r_e) = X^C y + t$ ·

1) Rs and RE are random, the former is equivalent to key, the latter is equivalent to noise;

2) T,a belong to known and the whole process does not change;

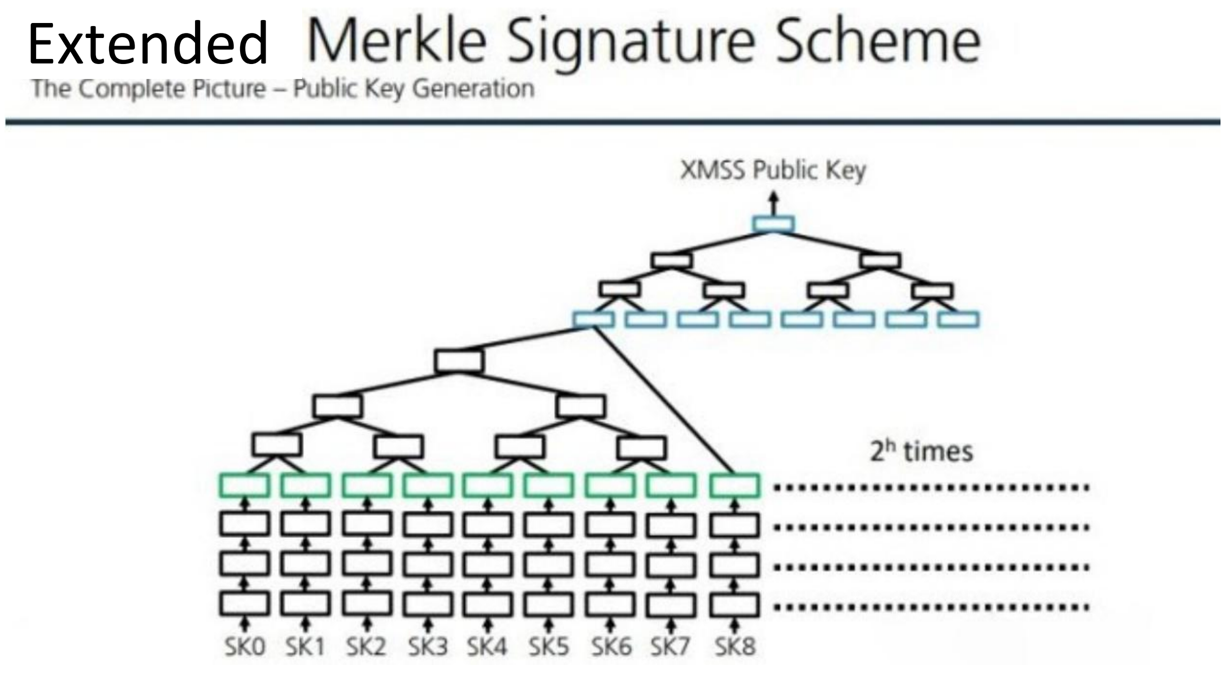
3) The function of C from V square is equivalent to a random variable, which is uncertain. The idea is that if P squared proves successful, the equation should be true no matter what parameters are passed from V squared.

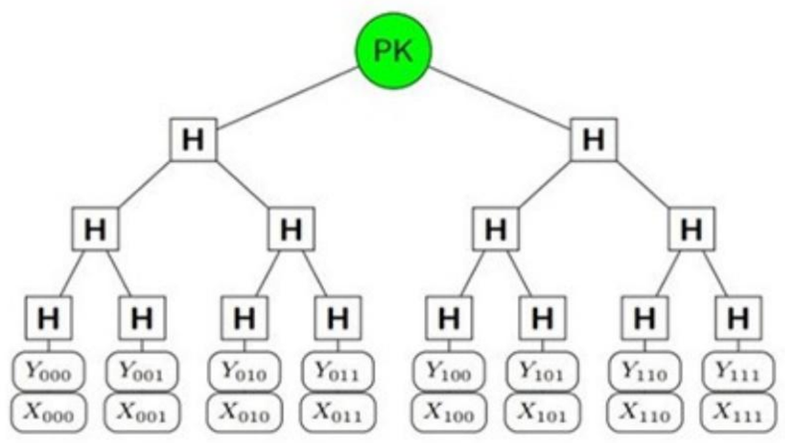
Lattice cipher is widely concerned by cryptographers, not only because it can resist quantum computing attacks, but also because lattice can construct some complex and powerful cryptographic applications, including many functions that traditional cryptosystems cannot achieve. First of all, fully homomorphic encryption (FHE) is the most typical example. In addition to all-homomorphic encryption, encryption based on lattice cryptography also includes functional encryption and obfuscation.

In the research on the post-quantum algorithm project of The National Institute of Standards and Technology (NIST), Dilithium is a Fiat Shamir signature constructed based on the MLWE problem. One of its characteristics is that efficient and concise uniform sampling is used in the signature algorithm. Dilithium signature scheme is constructed on the general lattice. In order to obtain a more compact public key size, Dilithium compresses the public key. NTRU lattice, on the other hand, on the password solutions than those of the general case the password on the efficiency and size parameters has more advantages, this paper gives the Dilithium signature on NTRU lattice of an efficient varieties, on the basis of inheriting Dilithium concise design, a combination of NTRU and rejection sampling technology advantage without additional compression processing, further enhance the efficiency of Fiat - Shamir signature based on the lattice.

2.3.3.2 Hash based

Hash-based signature algorithm, proposed by Ralph Merkel, is considered as one of the feasible alternatives to traditional digital signatures (RSA, DSA, ECDSA, etc.). The signature algorithm based on hash evolves from the one-time signature scheme and uses Merkle's hash tree authentication mechanism. The root of the hash tree is the public key, and the one-time authentication key is the leaf node in the tree. The security of signature algorithm based on hash depends on the collision resistance of hash function. Since there is no efficient quantum algorithm to quickly find collisions of hash functions, hashing based constructions can be resistant to quantum computer attacks. Furthermore, the security of hashing - based digital signature algorithms does not depend on a particular hash function. Even if some of the hash functions currently in use are breached, the compromised hash function can be directly replaced with a more secure hash function.


Extended Merkle Signature Scheme
The Complete Picture – Public Key Generation
XMSS Public Key
2h times
SK0
SK1
SK2
SK3
SK4
SK5
SK6
SK7
SK8


PK
H
Y_{000} Y_{001} Y_{010} Y_{011} Y_{100} Y_{101} Y_{110} Y_{111}
X_{000} X_{001} X_{010} X_{011} X_{100} X_{101} X_{110} X_{111}

Merkle Trees

- Merkle, 1979: Leverage one-time signatures to multiple messages
- Binary hash tree on top of OTS public keys

One-off Signature (OTS)

The hashing based signature scheme benefits from the Lamport OTS (one-time signature) scheme. The logic behind the Lamport scheme is clear: the signer generates random value pairs that need to be signed for each bit, and the pair that forms the private key. The public key is formed by the hash of these values. Instead, to sign a message, the signer reads the message bit by bit and displays a value, each secret pair depending on the bit value. The verifier can then verify that the hash of all secret values is equal to the corresponding hash value in the public key. Although the Lamport OTS hash is considered fast, the key and signature sizes are relatively large. For example, if SHA256 is used as the underlying hash function, the public key is made up of 512 256-bit hash outputs (1 hash pair per bit), and the signature is made up of 256 secret values (256 bits per bit). If we sum up the key and signature data, they will take up 24.5KB, similarly, if we use the SHA512 algorithm, it will take approximately 98KB.

Further enhancement to the original algorithm can significantly reduce the key size. Currently, WOTS and its variants are considered to be the most effective methods for key and signature compression, while the Bleichenbacher and Maurer graphical solutions attempt to achieve the best efficiency in terms of the signature size and the number of hash functions evaluated per message bit.

As annotations, one of the main differences between the OTS methods is whether the required security assumptions are resistant to anti-hash functions and whether additional bitmasks are used. Currently, WOTS -t proved in QROM model is safe, it is considered the most promising candidate WOTS package, because only one additional seed values need to be together with the public key, is used to calculate the required a bitmask, and its security was not influenced by the influence of the "paradox" birthday, it also introduces all keyed hash function call, in order to prevent a second original multi-objective like attack. The latter results in a shorter public key and hash output size.

Minor and multiple signatures

While there are many ways to turn a one-time signature scheme into a multiple-signing scheme, one popular approach is to use the Merkle Authentication Tree. Using the Merkeltree, the total number of signatures that can be published is defined at the time the key is generated. The main advantages of this approach are its short signature output and quick verification, while the disadvantages are the relatively long key generation time and their statefulness.
Moving to a stateless less-signed scheme requires additional complexity and larger signature output. The HORS (and its extended HORST) scheme is the most commonly used multiple stateless signature scheme, such as SPHINCS. Multi-hash signature schemes can be built by combining the {one-time and less frequent} schemes described above, and they can be divided into two classes: stateful (such as XMSS, LMS) and stateless (such as SPHINCS, SPHINCS +, Gravity, Simpira, Haraka). Stateful schemes typically produce shorter signatures, but they require a mechanism to maintain state (which paths/keys have been used). On the other hand, the stateless solution starts with a moderately sized Merkle tree or tree layer at the top, instead of using OTS signature at the bottom, they use a less robust signature scheme. The latter allows them to pick indexes at random, so there is no need to track the path state. The disadvantage of stateless schemes is their signature size, for example, in sphincs-256, each signature would be 41KB in size.

Speed and security of hash functions

The underlying hash algorithm is clearly important for the overall security of the proposed project. Several factors affect the choice of the algorithm, including speed, security level, and availability; For example, what hardware capabilities can be applied to improve runtime performance. The first thing to establish, however, is that the algorithm needs to be resistant to post-quantum (PQ) attacks. Sha-2 and sha-3 algorithms support a variety of digest sizes, namely 224,256,384, and 512 bits. We observe this with the improved search speed provided by Grover algorithm, where the collision resistance can be reduced from half of the selected size to one-third. Thus, in the presence of large-scale quantum computers, the 384-bit versions of SHA-2 and SHA-3 algorithms will provide 128-bit collision protection. The 256-bit version provides only 85 bits of security.

In addition, we observed that quantum image attacks against the 256-bit versions of SHA-2 and SHA-3 algorithms could be accomplished through 2^153.8 and 2^146.5 code loops, respectively.

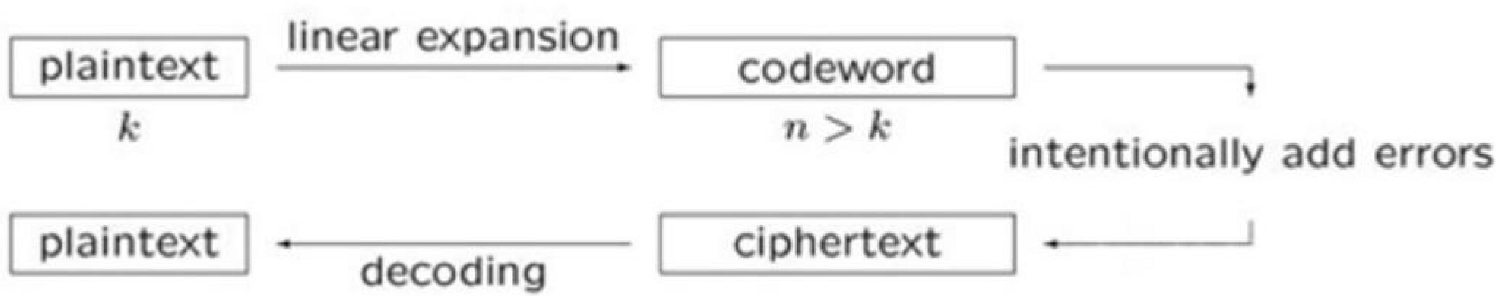
Based on these two observations, the SHA256 algorithm is not suitable for hashing security, but it is still relatively safe. It should also be noted that the post-quantum (PQ) algorithm is worse at cost performance than the classical Van Oorschot and Wiener hash collision algorithms, even under optimistic assumptions about the pace of quantum computer development.

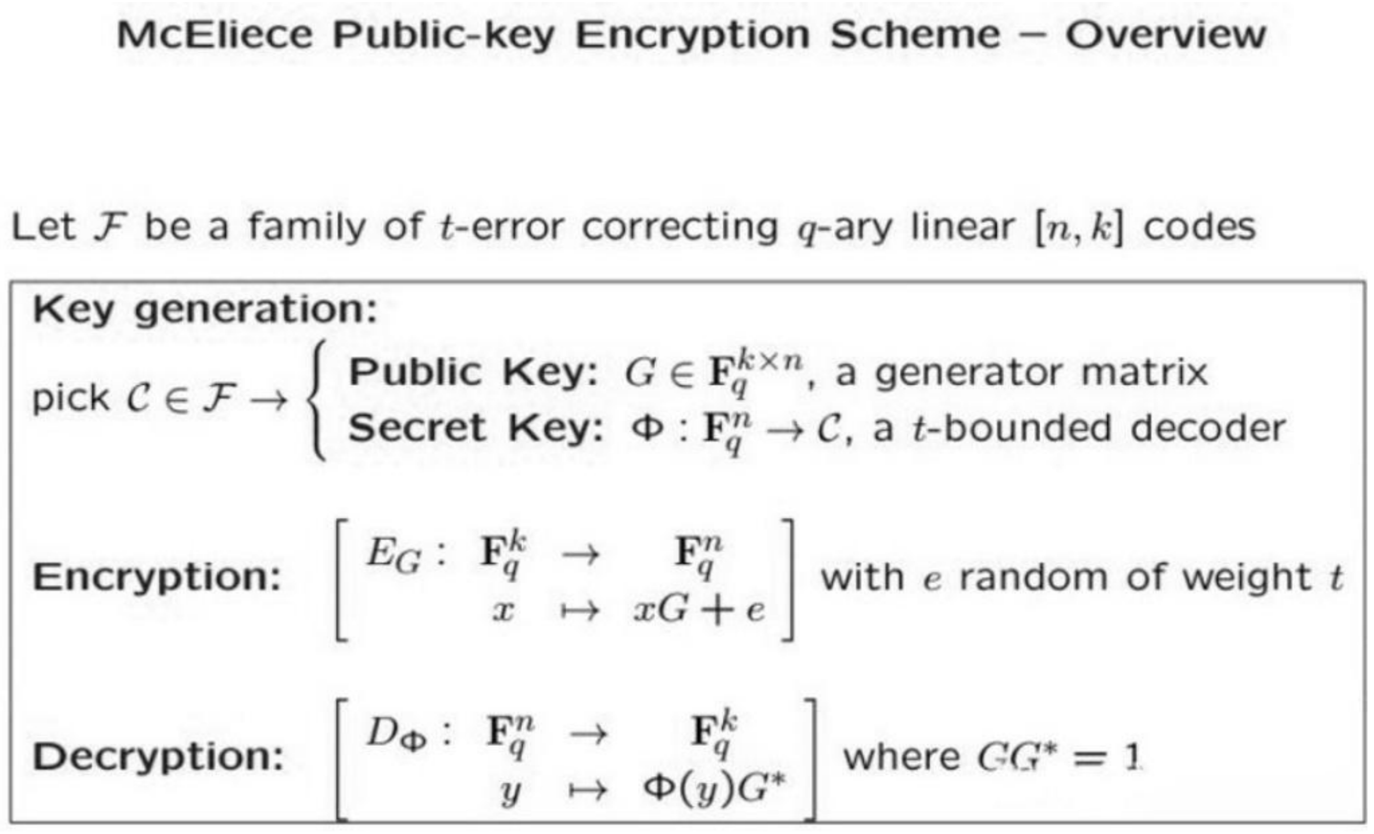
It is also important to note that different versions of SHA-3 often perform worse than their SHA-2 counterparts. One reason for this is that both SHA-1 and SHA-2 have more hardware support for modern processors. Note that although the SHA-2 algorithm does not provide protection against long scale attacks, it provides bit-security similar to that of SHA-3. Hashing based post-quantum signature schemes, including BPQS, are generally less prone to such attacks, so sha-2 will be considered for better performance advantages when compared with SHA-3. If performance is important, we can also consider the less supported BLAKE2b algorithm.

2.3.3.3 Code-based

The coding - based algorithm uses error correction codes to correct and calculate random errors. A well-known encode-based encryption algorithm is McEliece. McEliece uses the random binary irreducible Goppa code as the private key, and the public key is the general linear code after transforming the private key. Courtois, Finiasz and Sendrier used The Niederreiter public key encryption algorithm to construct the encoder-based signature scheme. The main problem with code-based algorithms such as McEliece is that the public key size is too large. Encoding - based algorithms include encryption, key exchange and so on.

Linear Codes for Cryptography





McEliece Public-key Encryption Scheme – Overview

Let $\mathcal{F}$ be a family of t-error correcting q-ary linear $[n, k]$ codes

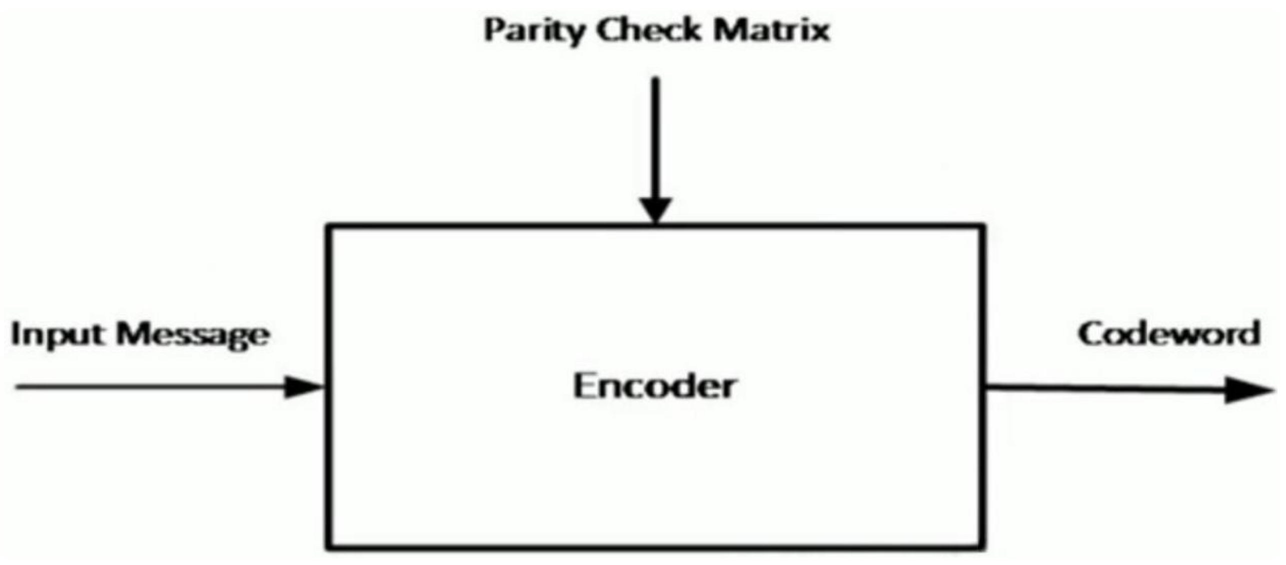
Key generation:

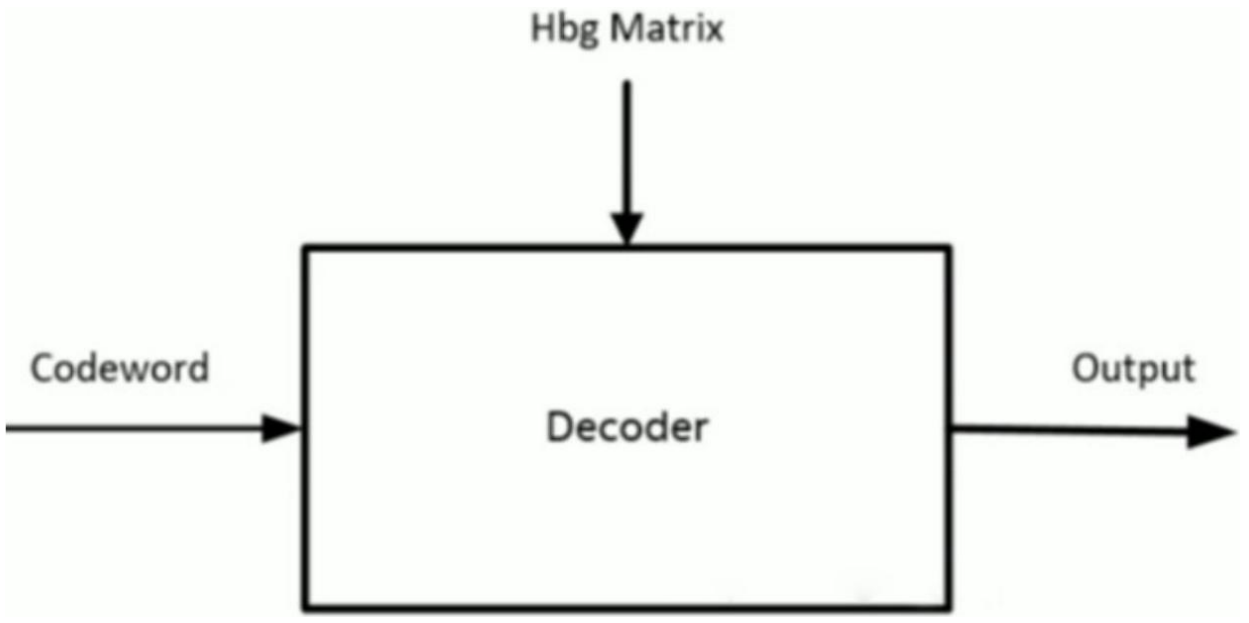
pick $\mathcal{C} \in \mathcal{F} \rightarrow \begin{cases} \textbf{Public Key: } G \in \mathbf{F}_q^{k\times n}, \text{ a generator matrix} \\ \textbf{Secret Key: } \Phi : \mathbf{F}_q^n \rightarrow \mathcal{C}, \text{ a } t\text{-bounded decoder} \end{cases}$

Encryption: $\left[\begin{array}{llll} E_G : & \mathbf{F}_q^k & \rightarrow & \mathbf{F}_q^n \\ & x & \mapsto & xG + e \end{array}\right]$ with e random of weight t

Decryption: $\left[\begin{array}{llll} D_\Phi : & \mathbf{F}_q^n & \rightarrow & \mathbf{F}_q^k \\ & y & \mapsto & \Phi(y)G^* \end{array}\right]$ where $GG^* = 1$

The simplified process is as follows: input information M and parity matrix, encoder is used for coding, and decoding is carried out after transmission through the channel. The decoding process is the process of signing, output signature and verification signature.





LDPC code

LDPC Code full name is the Low Density Check - Parity Code, put forward in 1962, is a kind of Check matrix of the sparse matrix of linear block codes, can greatly save the key storage space, BP decoding algorithm of LDPC codes can parallel implementation in hardware, greatly improve the decoding speed, BP decoding algorithm is put forward by Judea Pearl in 1982 for the first time, it is expressed as precise reasoning algorithm for tree, then extended to more trees. Although it is not accurate in general graphics, it has been proved to be a useful approximation algorithm. BP algorithm is a common method in artificial intelligence and information theory, and has proved successful experience in many applications, including low density parity check code, Turbo code, free energy approximation and satisfiability. The main idea is to make use of the information received during each iteration to carry out continuous information transfer and iterative operation between variable node and calibration node, so as to carry out decoding.

2.3.3.4 Based on multiple variables

The algorithm based on multiple variables uses the quadratic polynomial group with multiple variables on the finite field to construct encryption, signature, key exchange and other algorithms. The security of multivariable ciphers depends on the difficulty of solving nonlinear equations, that is, quadric polynomials with multiple variables. The problem proved to be a nondeterministic polynomial time difficulty. At present, there are no known classical and quantum algorithms to solve multivariable equations in the finite field quickly. Compared with the classical cryptography algorithm based on number theory, the multi-variable algorithm is faster in calculation and larger in public key size, so it is suitable for the application scenarios without frequent public key transmission.

Multivariate Cryptography [DS06]

MPKC: Multivariate Public Key Cryptosystem

Public Key: System of nonlinear polynomials

$$p^{(1)}(x_1,\cdots,x_n) = \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(1)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(1)} \cdot x_i + p_0^{(1)}$$

$$p^{(2)}(x_1,\cdots,x_n) = \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(2)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(1)} \cdot x_i + p_0^{(2)}$$

$$p^{(m)}(x_1,\cdots,x_n) = \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(m)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(m)} \cdot x_i + p_0^{(m)}$$

d: = degree of the polynomials in the system

m: = # equations

n: = # variables

3 The relationship between quantum computer and block chain

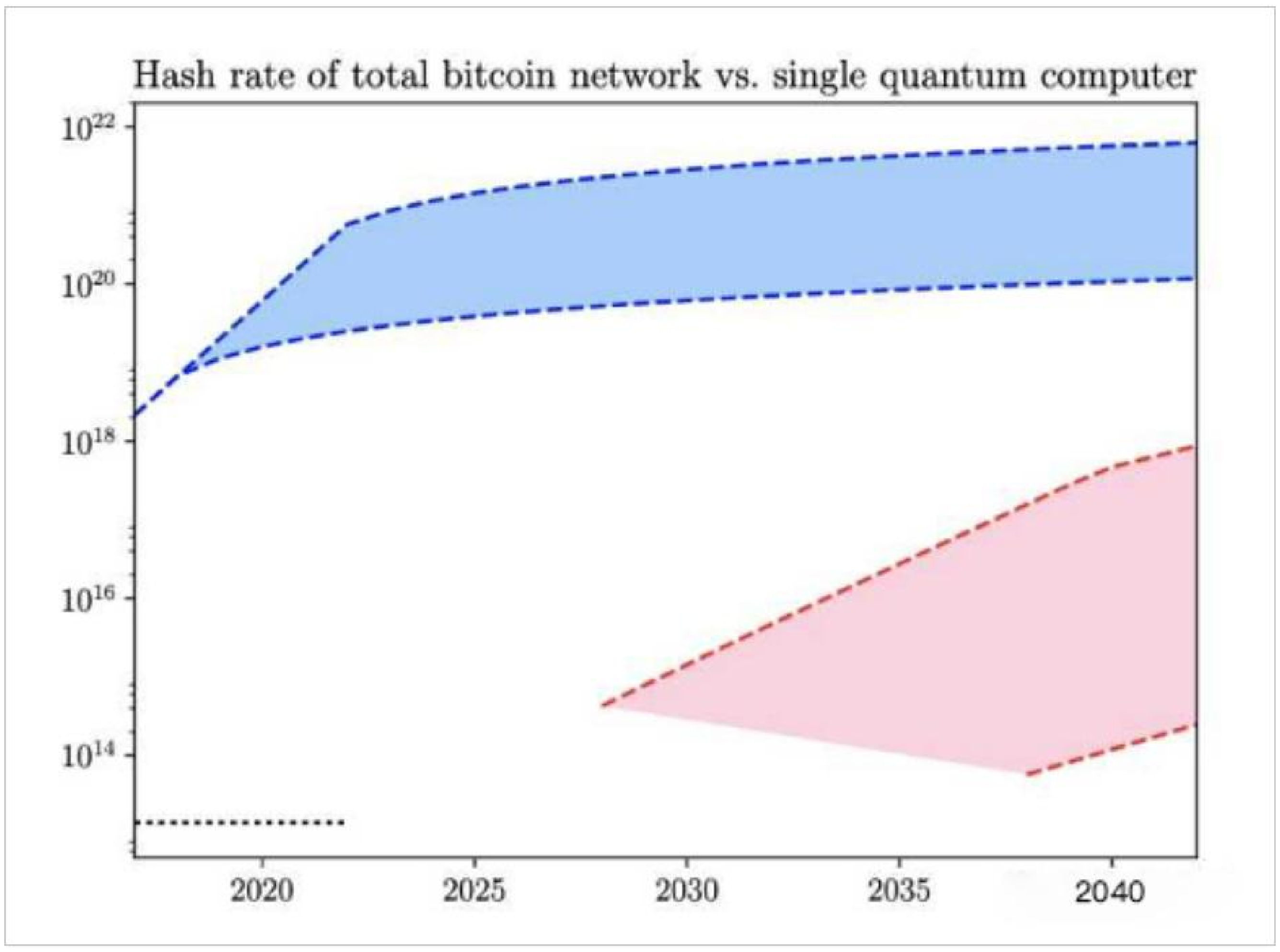
In traditional computers, RSA asymmetric encryption algorithm of traditional cryptography, etc. RSA asymmetric encryption algorithm is based on a large number to perform prime factorization, because it needs to calculate the prime factorization of a large number needs to use a lot of time, so due to the complexity of the reason, the password security degree is greatly guaranteed. In block chain technology, digital signature private key for trading asymmetric encryption USES the RSA asymmetric encryption algorithm, the core technology is the use of the elliptic curve algorithm (ECDSA) generated password key and a private key, you can easily calculate the corresponding public key, on the contrary, will be very difficult to calculate the private key, this seemingly safe one-way technology, is the foundation of the structure now block chain technology.

3.1 Influence of Quantum computer

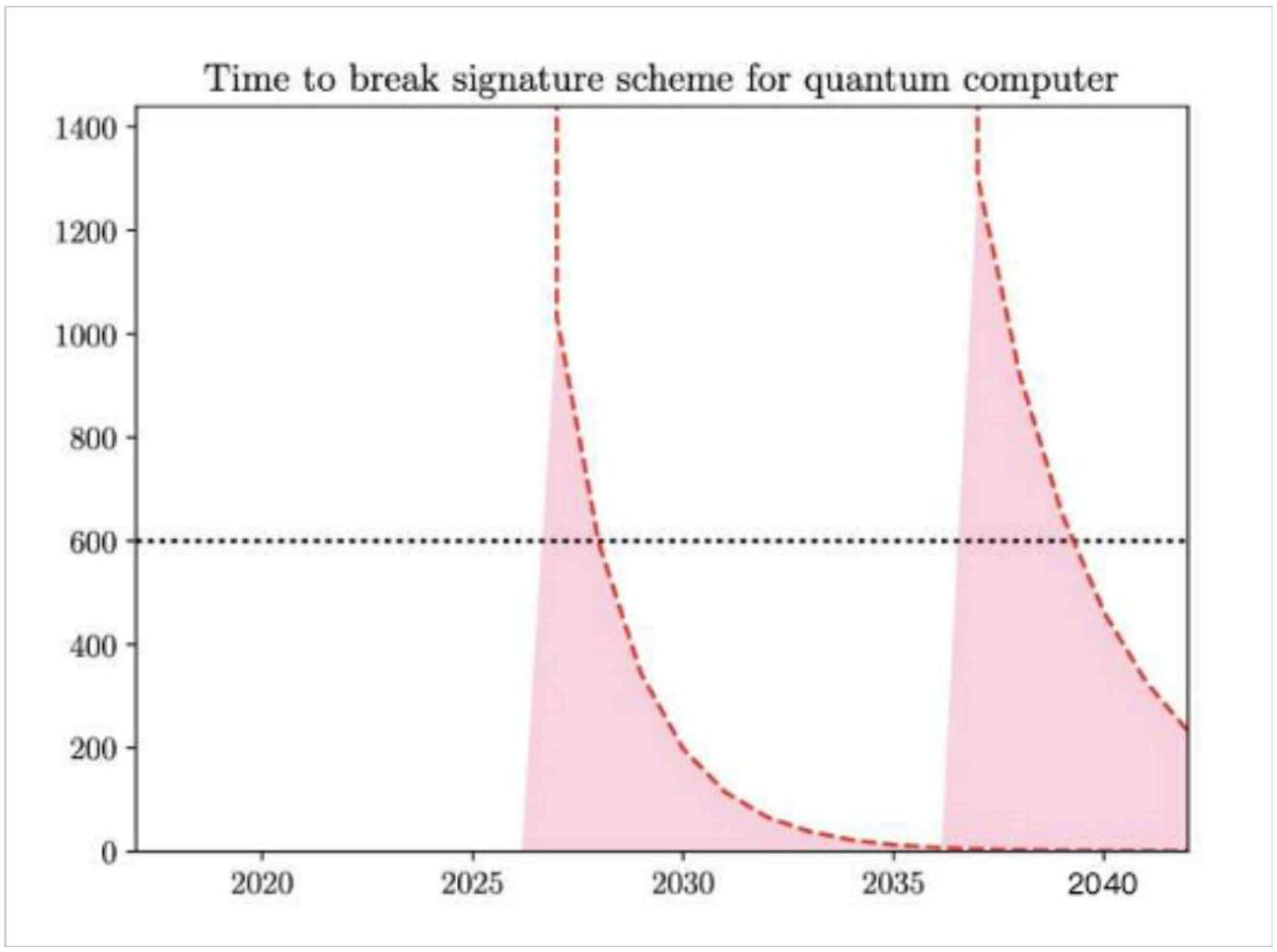
Block has been brought about by the value chain, to a large extent depends on the freedom and security, fairness, and the value is based on Hash function (Hash) and asymmetric encryption algorithm, existing in the traditional computer to calculate force is difficult to crack, this is also the cause of security, but, with the mature of quantum computing, calculate the force of the exponential increase, probably will lead to a dramatic change. Imagine if the core of blockchain technology, the private key, could be easily reversely deduced by quantum computers. What would happen to the application of blockchain technology and the systems it constitutes?

The continuous progress of quantum computing poses a challenge to the current blockchain technology. Fundamentally speaking, it is not only the blockchain technology, but also poses a serious threat to all the existing communication security systems in the world, and even subversions the existing industry. The current mainstream cryptography asymmetric encryption algorithm is based on a large number of decomposition of prime factors, and calculate method of security is based on the current conventional computers can't key in a reasonable amount of time calculated, so as to make the crack cost is much higher than be cracked the value of information itself, it also has mentioned to break down a 300 - bit prime factors, using traditional to use the computer in 150000, and using a quantum computer, just a second, in theory, quantum computing can achieve rapid decomposition of any large integer code, this will lead to asymmetric encryption algorithm in the world of quantum computing is vulnerable. In other aspects of traditional cryptography, some encrypted chat programs, SSL certificates, and some encrypted data storage would be weak if the private key could easily be pushed backwards.

So, if quantum computing has developed to the degree that the block chain technology in traditional cryptography encryption algorithm also cannot stand still, DMO is the resistance to the technical scheme and quantum has been research, if one day the RSA algorithm were declared to be unsafe, the existing block chain technology and will have the opportunity to use community DMO quantum encryption algorithm and consensus resistance mechanism to protect, not devastating effects on the block chain technology.



Since quantum computers don't have much advantage over classical computing in hashing problems, they don't pose much of a threat to mining.



Quantum computers, of course, to achieve the computing power of bits and logical 门 fault-tolerant rate is also a big challenge, if need to sign in 10 minutes break, in the gate under the error is also need to order of magnitude of quantum bit (physical bits), of course, if gate fideltiy to seven or eight will greatly reduce the number of bits is necessary, so the direction of development is still a long way to go. So given the potential for quantum threats to cryptocurrencies and blockchains in the future, is there a way to solve these problems?

4 About the DMO

DMO is a kind of based on lattice algorithm based on quantum encryption signature algorithm after signature of the chain, is the focus of the quantum algorithm after even a quantum computer to ensure the safety, quantum computers becoming mature, will be very effectively break based on number theory (RSA, DSA and diffie-hellman ElGamal and elliptic curve variation) of conventional asymmetric encryption and digital signature algorithm.

The DMO refers to some signature schemes proposed by Gentry, Peikert and Vaikuntanathan, etc., and uses a type door sampler called "Fast Fourier Sampling" to instantiate and improve the framework on the network. This signature scheme is called Monster signature.

In addition, in response to larger signature lengths and to achieve a new balance of efficiency and security, the DMO uses a new consensus mechanism that combines PoW and PoS -- PoT (Power of Trust).

4.1 Signature Scheme

The signature scheme of DMO uses Monster signature, which combines the advantages of anti-quantum signature algorithms such as CRYSTALS, Falcon, and Rainbow. The safety of DILITHIUM depends on the hardness of MLWE and module Short integer solution problem (MSIS), and follows Fiat-Shamir and abort technique. DILITHIUM by uniform distribution of all the parameters set and sample using the same module and ring, the realization of which makes it more simple than FALCON, DILITHIUM on key and signature size, and the key generation, signature and authentication efficiency of the algorithm, has the very strong and balanced performance, the FALCON is a kind of based on fast Fourier Ye Ge of NTRU compactification signature, based on the framework of GPV (constructed GPV08) encoding and decoding problem, the problem of potential difficulty is the problem of the shortest integer solutions on NTRU lattice (SIS), in the general case,Even with the help of quantum computer, there is no effective solution algorithm at present.

Falcon's high-level design was simple: It instantiated the theoretical framework described by Gentry, which required two components:

• A type of cipher. We chose a class of NTRU lattices.

• Trapdoor sampler. We rely on a new technique called fast Fourier sampling.

In short, it can be described as follows:

Falcon=GPV framework +NTRU lattice + Fast Fourier sampling.

Rainbow is a multi-variable signature scheme whose layered structure is based on an unbalanced oil-curate (UOV) signature scheme. Rainbow layer is imposed by the additional structure makes it face more password analysis technology, but improve the efficiency of the scheme, Rainbow signature and verification provides quick and very short signature, but it is a very large public key, however, due to the key size is very large, Rainbow is not suitable for as a general purpose signature algorithm to replace the current algorithm appears in the FIPS186-4. In particular, large public keys make certificate chains very large. However, some applications do not need to send keys frequently.

Through the statistical analysis found that, based on lattice structure and encoding is one of the most, and is mainly used to construct the public key encryption (secret key exchange) algorithm, because the trapdoor structure based on multivariate relative more feasible and efficient, so based on multivariate structure is mainly focused on digital signature scheme, public key encryption scheme is less, because of the construction scheme based on the hash tree structure in use, so only the structure of the digital signature, the lack of public key encryption algorithm.

Specific implementation code:

```
Gen

01. A ← $R_q^{k\times\ell}$
02. $(s_1, s_2) \leftarrow s_\eta^\ell \times s_\eta^k$
03. t:= $As_1 + s_2$
04. return (pk = (A, t,),sk =(A,t,$s_1$ $s_2$))
Sign(sk/M)
O5. Z: = ⊥
06. while Z = ⊥ do
07.     y ← $S^\ell_{yi-1}$
08.     $W_1 := HighBits(Ay, 2_{y2})$
09.     c ∈ $B_{60}$:=$H(M||w_1)$
10.     Z:= y + $cs_1$
11.     if ||Z||∞ ≥ $y_1 - \beta$ or $||LowBits(Ay - sc_2, 2_{y2})||\infty \geq y_2 - \beta$, then Z := ⊥
12. return $\sigma = (z,c)$
Verify (pk, M,$\sigma = (z,c)$)
13. $w'_1 := HighBits(Az - ct, 2_{y2})$
14. if return $[\![ ||z||\infty < y_1 - \beta ]\!]$ and $[\![ c = H||w'_1 ]\!]$
```

4.4.1 Key

Output public and private keys SK, PK

Algorithm 4 Keygen ($\emptyset$,q)

Require: A monic polynomial$\emptyset$ $\mathbb{Z}[x]$, $a\ modulus\ q$

Ensure: A secret key sk, a public key pk

```
1: $f,g,F,G, \leftarrow NTRUGen(\emptyset, q)$
2: B$\leftarrow \left[\frac{g}{G}\middle|\frac{-f}{-F}\right]$.                ▻Solving the NTRU equation
3: $\widehat{B} \leftarrow FFT(B)$          ▻ Compute the FFT for each of the 4 components {g,-f,G, -F}
4: G$\leftarrow \widehat{B} \times \widehat{B} *$
5: T$\leftarrow ffLDL^{*}(G)$                                  ▻ Computing the $LDL^{*}$ tree
6: for each leaf leaf of T do                                  ▻ Normalization step
7: |   leaf. value $\leftarrow \not\phi$  $\sqrt{\text{leaf. value}}$
8: sk $\leftarrow (\widehat{B},T)$
9: $h \leftarrow g\, f^{-1} mod\ q$
10: $pk \leftarrow h$
11: return sk, pk
```

Repeat a few times until the signature is produced, and we attach a counter so that each signature attempt for the same message is different from the Shake-256 output.

Also, since each message may require multiple iterations to be signed, we use a conflict-resistant hash function to calculate the initial digest of the message and use this digest instead of the message throughout the signature process.

4.1.2 Signature

Algorithm 10 Sign (m,sk ,$\lfloor\beta^2\rfloor$)

Require: A message m, a secret key sk a bound $\lfloor\beta^2\rfloor$

Ensure: A signature sig of m

```
1: $r \leftarrow \{0,1\}^{320}\ uniformly$
2: c$\leftarrow HashToPoint\ (r||m,q,n)$
3: t$\leftarrow \left(-\frac{1}{q}FFT(c) \odot FFT\ (F), \frac{1}{q}FFT(c\ ) \odot\ FFT\ (f)\right)$        ▻ $t = (FFT(c)FFT(0))\ \cdot\ \widehat{B}^{-1}$
```

```
4: do
5: |  do
6: |  |  Z ← $ffSampling_n(t,T)$
7: |  |  s=(t-z)$\widehat{B}$        ▻ $At\ this\ point,\ s\ follows\ a\ Gaussian\ distribution: s \sim D_{(c,0)+\wedge(B),\sigma,0}$
8: |  While $\|s\|^2 > \lfloor\beta^2\rfloor$        ▻ $Since\ s\ is\ in\ FFT\ representation, one\ may\ use\ (3.8)\ to\ compute\ \|s\|^2$
9: |  $(s_1,s_2) \leftarrow invFFT(s)$        ▻ $s_1 + s_2h = c\ mod\ (\emptyset\ ,\ q)$
10: |  $s \leftarrow Compress(s_2, 8 \cdot sbytelen - 328\ )$        ▻ $Remove\ 1\ byte\ for\ the\ header, and\ 40\ bytes\ for\ r$
11: while (s = ⊥)
12: return sig = (r,s)
```

Description:

1）The Hash.

2）Photosampling →\rightarrow→ (s1,s2) \rightarrow(s_1,s_2)→(S1, S2);

3）Compress S 2, s_2, S2, Compress(S 2)(s_2)(s2);

4）Signature sigma (r, s) \ sigma (r, s) sigma (r, s).

The most subtle part of the signature algorithm is fast Fourier sampling, because it uses discrete Gaussian functions over falcon trees and Z. The rest of the algorithm, including signature compression, is fairly easy to implement. Formally, given the private key SK and message M, the signer uses SK to sign the MAS, as shown below:

1）A random SALT is generated uniformly in, and then the concatenated string is hashed;$\{0,1\}^{320}$(r||m)

2）A (not necessarily short) pre-image of C is calculated and then used as input to the FAST Fourier sampling algorithm specified by ffSampling;

3）s_2Encoded (compressed) for the bit string S specified in the compression;

4) The signature consists of a pair of (R, s).

4.1.3 Validation

Algorithm 16 Verify (m, sig, pk, $\lfloor\beta^2\rfloor$)

Require: A message m, a signature sig = (r, s), a public key pk = h$\in \mathbb{Z}_q[x]/(\emptyset)$, $a\ bound\ \lfloor\beta^2\rfloor$

Ensure: Accept or reject

```
1: c ← Hash To Point (r||m, q, n)
2: $s_2$ ← $Decompress\ (s, 8 \cdot sbytelen - 328)$
3: if ($s_2 = \perp$) then
  4: |   reject.                                   ▻Reject invalid encodings
  5: $s_1 \leftarrow c - s_2 h\ mod\ q$            ▻ $s_1$ should be normalized between $\left\lceil -\frac{q}{2} \right\rceil$ and $\left\lfloor \frac{q}{2} \right\rfloor$
  6: if $\|(s_1, s_2)\|^2 < \lfloor\beta^2\rfloor then$
  7: |   Accept
  8: else
  9: |   reject                                    ▻ Reject signatures that are too long
```

Description:

1) Hash message and Salt;

2) Decoding signature;

3) Calculate S1, s_1, S1;

4) Norm verification.

Note: This rejection is not a rejection of the sample. The rejection of the sample is used during the signature phase.

The signature verification process is much simpler than key generation and signature generation. Given the public key PK =h, message M, signature SIg = and accept boundary, the verifier uses PK to verify that SIg is a valid signature of message M, as described below:$(r,s)[\beta^2]$

1) connect the assigns (called "salt") and the message m to a string (r||m), which is hash $\in$ Z, [x]/ (), specified by hashed to the point;

2) S decoding (decompression) polynomial /;$s_2 \in \mathrm{Z}[x](\varphi)$

3) Calculation value = c-H mod Q;$s_1 s_2$

4) If [], the signature is considered valid.$||(s_1,s_1)||^2 \leq \beta^2$Otherwise, they will be rejected.

4.2 Related Technologies

The basic methods of cryptosystem design are diffusion and obfuscation in order to resist the statistical analysis of the cryptosystem by the adversary.

Diffusion: To Diffusion the statistical properties of the plaintext into the ciphertext by making each bit of the ciphertext be generated by the multiplicity of the plaintext.The statistical characteristics of plaintext are spread out, so the probability of each letter in ciphertext is closer to equal, which makes it difficult for opponents to get useful information through statistical analysis.

Confusion: To make the statistical relationship between the ciphertext and the key as complicated as possible so that the enemy cannot obtain the key.Even if the enemy gets the statistical relationship between ciphertexts, it is difficult to get the key.

A practical cryptographic system must meet the following basic requirements:

1) The system should be practically secure, and when intercepting ciphertext or known plaintext-ciphertext pairs, it should be determined that the key or any plaintext is computatively infeasible.

2) Encryption and decryption algorithm is applicable to all elements in the key space.

3) The security of data depends on the key rather than the confidentiality of the algorithm.

4) The system is easy to implement and simple to use, and the efficiency of communication network should not be excessively reduced.

The consensus framework for defining DMO simplifies Dilithium integration. The open source code of the scats-Dilithium digital signature scheme is added to the smart module, and some minor modifications are made.

One of the changes is to allow users to create new types of quantum secure addresses. First, the user registers the handle of their choice. The registration process executes transactions on the blockchain that include the user's quantum-secure public key and selected handle. This allows only 32 bytes of data (the transaction ID that registered the transaction) to cover 3 kilobytes of data (the uncompressed Dilithium public key).

More importantly, this will allow users to send money to each other's handles. Rather than typing a normal address with random numbers and letters, Dilithium users simply type the handle into the GUI and click send. If James' JL777 'Lee chooses to use the "JL777" handle and register it with his Dilithium public key, then you can send the money to "JL777" and the money will be transferred directly to the other party's wallet in a quantum secure manner.

The Dilithium smart module also has another major feature, adding an additional consensus rule that requires each transaction to be signed twice: once according to the original digital signature process of the blockchain, and then a second time according to Dilithium's new anti-quantum signature process. This merely ensures that any transactions on the blockchain using the Dilithium module are protected from quantum computer attacks.

Since Dilithium is encoded using the DMO's definition consensus framework, quantum secure trading is not unique to a blockchain. In contrast, the Dilithium module can be used for any project built within the DMO ecosystem. Quantum security can be simply added to any chain as a plug-in, and Dilithium definition consensus module integration is just one example of a DMO.

4.2.1 Basic definitions

N dimensional Lattice L is any subset that satisfies the following two conditions:R^n

1) In an additional subgroup: 0∈L, −x, x + y∈L and x, y∈L,

2) Dispersion: Every x∈L has a neighborhood in R, where X is a unique lattice point.R^n

For example, there are integer lattices and scaled down lattices cL for any real number C and lattices L, andZ^n

Checkerboard lattice {x∈:}.$\mathrm{Z}^n \sum_i x_i$

The minimum distance of lattice point L is the length of the shortest non-zero lattice vector is:

λ_1(L):= (||·|| represents the Euclidean norm).$_{v \in L\setminus\{0\}}^{\min}||v||$

The ith continuous minimum (L) is the smallest r, so L has at most I linearly independent norm vectors r.λ_i

Since lattice L is an additive subgroup of R, we have a superposed group of quotient group /L:$\mathrm{R}^n\mathrm{R}^n$

C + L = {c + v: v∈L}, c∈,R^n

In contrast to the usual inductive addition operation (+L)+(+L)=(+)+ L, the basic domain of an L is a set F∈ which contains a representative ∈(C +L) ∩F for each coset of C +L.$c_1 c_2 c_1 c_2$, $c^\#$

Although each (unstructured) lattice L is infinite, it can generate an integer linear combination of a finite number of linearly independent basis vectors.b_1,} :b_k

L = L (B) : = B = {},$\mathrm{Z}^k \sum_{i=1}^{k} z_i b_i$, $z_i \in \mathrm{Z}$

The integer k is the base permutation, is the invariant of the lattice, and we restrict the full permutation k to be equal to n. For any matrix U∈, the lattice B is not unique B dot U is also a basis for L(B), because U dot =.$Z^{n*\ n}$(即行列式为 ± 1)$Z^n Z^n$

For a lattice L based on B, a common fundamental domain is the fundamental domain centered at the origin, parallelepiped P(B):= B·, where coset C +L is the representative C-b ·.$[-\frac{1}{2}, \frac{1}{2})^n$ $[B^{-1}c]$

The dual lattice: an L⊂ the dual (sometimes called symmetric) lattice which is defined forR^n

L^* : = {w: ⊆ Z},$\langle w,L \rangle$

That is, the set of points whose inner product with the vector in L is all integers, which is easy to verify as a lattice, so it will not be repeated here.L^*

4.2.2 Short integer solutions

We now explain the most common computational problem of lattice in cryptography -- the short integer solution (SIS) problem, which is also the core of constructing Monster signature algorithm:

The short integer solution (SIS) problem was first introduced in Ajtai [Ajt96] as a basic one-way and collision resistant hash function.

The SIS problem requires that a large finite additive group of many uniformly random elements be given in order to find a fully subdivided "short" non-trivial integer combination whose sum is 0.

According to the definition, SIS problem is related to the worst lattice point problem, and the range of short integer solution decreases with the range of normal form constraint function of N-dimensional lattice. After polynomial time reduction, SIS can effectively solve the GapSVP problem because the result is closer to the average value than the high probability number. When SIVP is carried out on any n-dimensional lattice, the difficulty in obtaining such a reduction is to generate a uniform random SIS instance of the solution to find the short vectors of any lattice.

Monster takes the problem of SIS a step further by adopting R-SIS. In the worst lattice problem environment, R-SIS is similar to SIS. However, in the underlying lattice problem, R-SIS is specialized in algebraic structure lattices, called ideal lattices, which are generated from ring R, and thus play an important role in the safety characteristics of R-SIS and in the quantitative prevention of the potential worst case.

4.2.3 NTRU

In 1996, Hoffstein,Pipher, and Silverman[HPS98] designed the public key encryption scheme NTRU(also known as NTRUEncrypt) as part of their concurrent work with Ajtai (which was not disclosed until early 1998). This is the first cryptographic construct using polynomial rings and is also the most efficient solution to algebraic structure lattices. The NTRU cipher system is very efficient and the key is very compact, it withstands a lot of cryptanalysis and attempts to crack, and it is very easy to parameterize. In particular, there is no standard for simplifying the NTRU problem from the worst lattice problem to the NTRU problem, which means that it is almost impossible to crack the NTRU problem and get a password. The algorithm flow is as follows:

Key Generation

1: Choose f$\in T_{d+1,d,}\ g \in T\ (d,d)$;

2: $f_q = f^{-1} \in R_q; f_p = f^{-1} \in R_p$

3: Secret key: (f, f_p);

4: Public key: $h = f_q g \in R_q$;

Encryption

1: Plaintext m $\in R_q; Random\ r \in T\ (d, d)$;

2: Ciphertext: c = $pr \cdot h + m\ (mod\ q)$;

Decryption

1: $f \cdot e\ (mod\ q)$;

2: Centerlift to a ∈ R;

3: Output: b = $f_p \cdot a\ (mod\ p)$;

Such as:

(n, p, q, d) = (3, 3, 23, 1)

- First verify whether q ＞(6d + 1)p
- Alice chooses:

$$f(x) = 1 + x - x^2 (private\ key)$$

- Alice computes $g(x) = 1 - x$

$$f_q(x) = f(x)^{-1} \ \ (mod\ q)$$

$$= 12 + 12x^2 \ \ (mod\ 23)$$

$$f_q(x) = f(x)^{-1} \ \ (mod\ p)$$

$$= 2 + 2x \ \ (mod\ 3)\ (Private\ key)$$

- Public key:

$$h(x) = f(x)^{-1} \ \ \ (mod\ q)$$

$$= 11x + 12x^2 \ \ (mod\ 23)$$

- Bob decides to send Alice the message m(x) = -1 + x + x^2

  using the ephemeral key: r(x) = x - x^2

- Ciphertext:

$$c(x) = pr(x) \cdot h(x) + m(x) \ \ \ (mod\ q)$$

= 2 + 11x + 11x^2 (mod 23)

- Let n = 5 and q = 7, and consider the polynomial

$$a(x) = 5 + 3x - \ 6x^2 + \ 2x^3 + \ 4x^4 \ \in \ R_7$$

- The coefficients of the centered lift of a(x) are chosen from

{-3, -2, ..., 2, 3,},so

- Centered Lift of

$$a(x) = -2 + 3x - \ x^2 + \ 2x^3 + \ -3x^4 \ \in \ R.$$

The NTRU algorithm is implemented with the following parameters:

(N, P, q) = (7, 2, 29), Alice's public key is,

h(x)=23+23x+23+23+23+23+23$x^2x^3x^4x^5x^6$

Decryption of ciphertext with the private key to verify that it is equal to plaintext.

```
package main

import "fmt"

var N int = 7                                                                   //N

var P int = 2 var                                                               //P

Q int = 29                                                                      //Q

var poly_h = [7]int{23, 23, 23, 24, 23, 24, 23} // var poly_f = [7]int{1, 1, 1, 0, 1, 1, 0} var poly_f_P = [7]int{1,

0, 0, 0, 0, 1, 1} //poly_f var poly_m = [7]int{3, 0, 3, 0, 0, 0, 0}   // var poly_r = [7]int{1, 1, 0, 1, 0, 0, 1}   // var

temp_r [7]int                    //P*r(x)

var  poly_cipher  map[int]int  var  temp_m                                  //temp_r(x)*poly_h(x)+poly_m(x)

//F(x)*poly_cipher(x)

map[int]int var expon_cipher map[int]int

var decrypt_m map[int]int                                                 //x

func encrypt() { */ var i int var j int

var k int = 0

for i = 0;   i < 7;   i++ { /*P*r(x),temp_r*/

temp_r[i] = P * poly_r[i]                                                 //fmt.Println(poly_h[i]) }

poly_cipher = make(map[int]int)

expon_cipher = make(map[int]int)

for i=0; i;< 7i++{/*for,temp_poly_h(x)+poly_/}r(x)*  m(x)*

for j = 0;   j < 7;   j++ {

poly_cipher[k] = temp_r[i] * poly_h[j]                              //fmt.Print(poly_cipher[1])

if k > 6 {

expon_cipher[k] = (i + j) % 7
switch expon_cipher[k] {

case 0:

poly_cipher[0] += poly_cipher[k]                                //fmt.Println(poly_cipher[0])

case 1:

poly_cipher[1] += poly_cipher[k]

case 2:

poly_cipher[2] += poly_cipher[k]

case 3:

poly_cipher[3] += poly_cipher[k]

case 4:

poly_cipher[4] += poly_cipher[k]

case 5:

poly_cipher[5] += poly_cipher[k]

case 6:

poly_cipher[6] += poly_cipher[k] default:

fmt.Printf                                                          //fmt.Println(expon_cipher[k]

)

k++

}                                                                    //fmt.Println("\n")

poly_cipher[i]+=poly_m[i]/*+m(x)*//fmt.Println(poly_m[i])//poly_cipher[i]=poly_cipher[i]%Q//fmt.Println(

poly_cipher[i])}                                                       //fmt.Println(poly_cipher[0]

)

for i = 0;   i < 7;   i++ {

/*Q=29*/ poly_cipher[i] = poly_cipher[i] % Q              //fmt.Println(poly_cipher[i]

)}                                                                   //fmt.Println(k)

}

func decrypt() {

var i int

var j int

var k int = 0

temp_m = make(map[int]int)
```

```
for i = 0;   i < 7;   i++ { /*poly_f(x)*poly_cipher(x)*/

//fmt.Println(poly_cipher[1])

for j = 0;   j < 7;   j++ {

temp_m[k] = poly_f[i] * poly_cipher[j] //fmt.Println("location=", k) //fmt.Println( temp_m[k])

if k > 6 {

expon_cipher[k] = (i + j) % 7

switch expon_cipher[k] {

case 0:

temp_m[0] += temp_m[k]                                        //fmt.Println(temp_m[1])

case 1:

temp_m[1] += temp_m[k]

//fmt.Print(expon_cipher[k],i,j,temp_m[k], temp_m[1]) //fmt.Println("\n")

case 2:

temp_m[2] += temp_m[k]

case 3:

temp_m[3] += temp_m[k]

/*fmt.Print( expon_cipher[k], temp_m[k], k, i, j, temp_m[1]) fmt.Println("\n")*/

case 4:

temp_m[4] += temp_m[k]

case 5:

temp_m[5] += temp_m[k]

case 6:

temp_m[6] += temp_m[k] default:

fmt.Printf

}}

k++

}}

//fmt.Println(k)

for i=0; i7;<i++{/*Q=29*/

temp_m[i] = temp_m[i] % Q //fmt.Println(temp_m[i])

}
```

```
var half_Q int = Q / 2

for i=0; i7;<i++{/*center lift*/

if temp_m[i] > half_Q && temp_m[i] > 0 { temp_m[i] = temp_m[i] - Q

fmt.Println(temp_m[i])

} else if temp_m[i] < -half_Q && temp_m[i] < 0 { temp_m[i] = temp_m[i] + Q

}}

decrypt_m = make(map[int]int)

k = 0

for i = 0;   i < 7;   i++ { /*poly_f_P(x)*temp_m(x)*/

//fmt.Println(poly_cipher[1])

for j = 0;   j < 7;   j++ {

decrypt_m[k] = poly_f_P[i] * temp_m[j] //fmt.Println("location=", k)

//fmt.Println(decrypt_m[k])

if k > 6 { /

expon_cipher[k] = (i + j) % 7 switch expon_cipher[k] {

case 0:

decrypt_m[0] += decrypt_m[k]

case 1:

//fmt.Println(decrypt_m[1])

decrypt_m[1] += decrypt_m[k]

//fmt.Print(expon_cipher[k], i, j, decrypt_m[k], decrypt_m[1]) //fmt.Println("\n")

case 2:

decrypt_m[2] += decrypt_m[k]

case 3:

decrypt_m[3] += decrypt_m[k]

/*fmt.Print(expon_cipher[k], decrypt_m[k], k, i, j, decrypt_m[1]) fmt.Println("\n")*/

case 4:

decrypt_m[4] += decrypt_m[k]

case 5:

decrypt_m[5] += decrypt_m[k] case 6:

decrypt_m[6] += decrypt_m[k] default:
```

```
fmt.Printf
}}
k++
}}
//fmt.Println(k)
for i = 0;   i < 7;   i++ {decrypt_m[i] = decrypt_m[i] % P fmt.Println(decrypt_m[i])
}}
func main() { encrypt() decrypt()
}
```

4.2.4 Fast Fourier sampling

Fast Fourier sampling is a fast algorithm of the Discrete Fourier Transform, which is obtained by improving the algorithm of the Discrete Fourier Transform according to the odd, even, virtual, real and other characteristics of the Discrete Fourier transform. It is not new to the Fourier transform theory, but it is a big step forward in the application of the Discrete Fourier transform in computer or digital systems. A complex sequence of x (n) for n items, by the DFT transform, either x (m) the calculations need n complex multiplication and n - 1 the plural addition, and a complex multiplication is four times real multiplication and two real addition, a complex additive is equal to two real addition, even if a complex multiplication and a complex additive defined as a "arithmetic" (four real multiplication and four real addition), then calculate n complex sequence x (m), the n point DFT calculations about commutation needs.N^2When N = 1024 points or more, need N2 = 1048576 calculations, in FFT, the use of WN cyclical and symmetry, a N a sequence (N = 2 k, k is positive integer), divided into two N / 2 items of sub sequence, each N / 2 point DFT transform need (N / 2) two operation, then the two N / 2 with N operation point DFT transform group of synthetic a N point DFT transform. So when you do that, the total number of operations becomes N plus 2 times N over 2 is equal to N plus N2 over 2. Continuing with the example above, at N=1024, the total number of operations becomes 525,312, saving about 50% of the computation. And if we are going to the ideology of "split" constantly, until the DFT is divided into two group of computing unit, then the point N of DFT transformation just Nlog2N time operation, N in 1024, the computational cost only 10240 times, is 1% of the previous algorithm directly, the more points, the computation of the save the more big, this is the superiority of FFT.

Fast Fourier language implementation, here is a fast Fourier code:

```
<span style="font-size:12px;" >
void fft(complex f[], int N){
complex t, wn; //
int i, j, k, la, lb, lc, l, r, m, n;
int M;
/*----M=nextpow2(N)----*/
For(i=N,M=1; (i=i/2)! = 1; M++);
For (I = 1, j = N / 2; iN-2;≤i++){
if (i<j){
t = f[j];
f[j]=f[i];
f[i]=t;
}
K=N/2
While (k j or less) {
j=j-k;
k=k/2; }
j=j+k}
/*----FFT----*/
for(m=1; M m or less.m++){
La = pow (2.0 m);   //la=2^m lb = la / 2;      //lb for (l = 1;   l <= lb;   l++)
lr = (l - 1)*pow(double(2), M - m);   for (n = l - 1;   n<N - 1;   n = n + la) //N/la{
Lc=n+lb; //n,lc
Wn_i(N,r,&wn,1); //wn=Wnr
c_mul(f[lc], wn, &t); //t = f[lc] * wn
c_sub(f[n], t, &(f[lc])); //f[lc] = f[n] - f[lc] * Wnr c_plus(f[n], t, &(f[n])); //f[n] = f[n] + f[lc] * Wnr
```

}}}}</span

Binary reverse order (8-bit reverse order)

```
uint8 bin8_rev(uint8 data){
data=((data&0xf0)>>4)|((data&0x0f)<<4);                    data=((data&0xCC)>>2)|((data&0x33)<<2);
data=((data&0xAA)>>1) |((data&0x55)<<1);
return data;
}
```

Binary reverse order (16-bit reverse order)

```
static INT16U BitChange(INT16U us_DataIn){
INT16U us_Data=us_DataIn;
us_Data=((us_Data&0xFF00)>>8)|((us_Data&0x00FF)<<8);
us_Data=((us_Data&0xF0F0)>>4)|((us_Data&0x0F0F)<<4);
us_Data=((us_Data&0xCCCC)>>2)|((us_Data&0x3333)<<2);
us_Data=((us_Data&0xAAAA)>>1)|((us_Data&0x5555)<<1);
return (us_Data);
}
```

Fourier Series

Fourier series tells us that the periodic signal can be decomposed into a finite or infinite sine or cosine wave superposition, and the frequencies of those waves are integer times of the original signal frequency, in a nutshell, is the driving force to do the Fourier transform (if is a driving force cycle are expanded in Fourier series), the special solution of each base driving force, and then superimposed to the particular solution, will be a function for the Fourier transform or expanded in Fourier series, can help us to solve the linear differential equation, or from the actual sense, can help us to analyze how a linear system to the outside world to make the response.

Any function can decompose the sum of odd even functions;

f(x)=[f(x)+f(-x)]/2+[f(x)-f(-x)]/2

Sine of x sine of x and cosine of x cosine of x are odd and even functions.

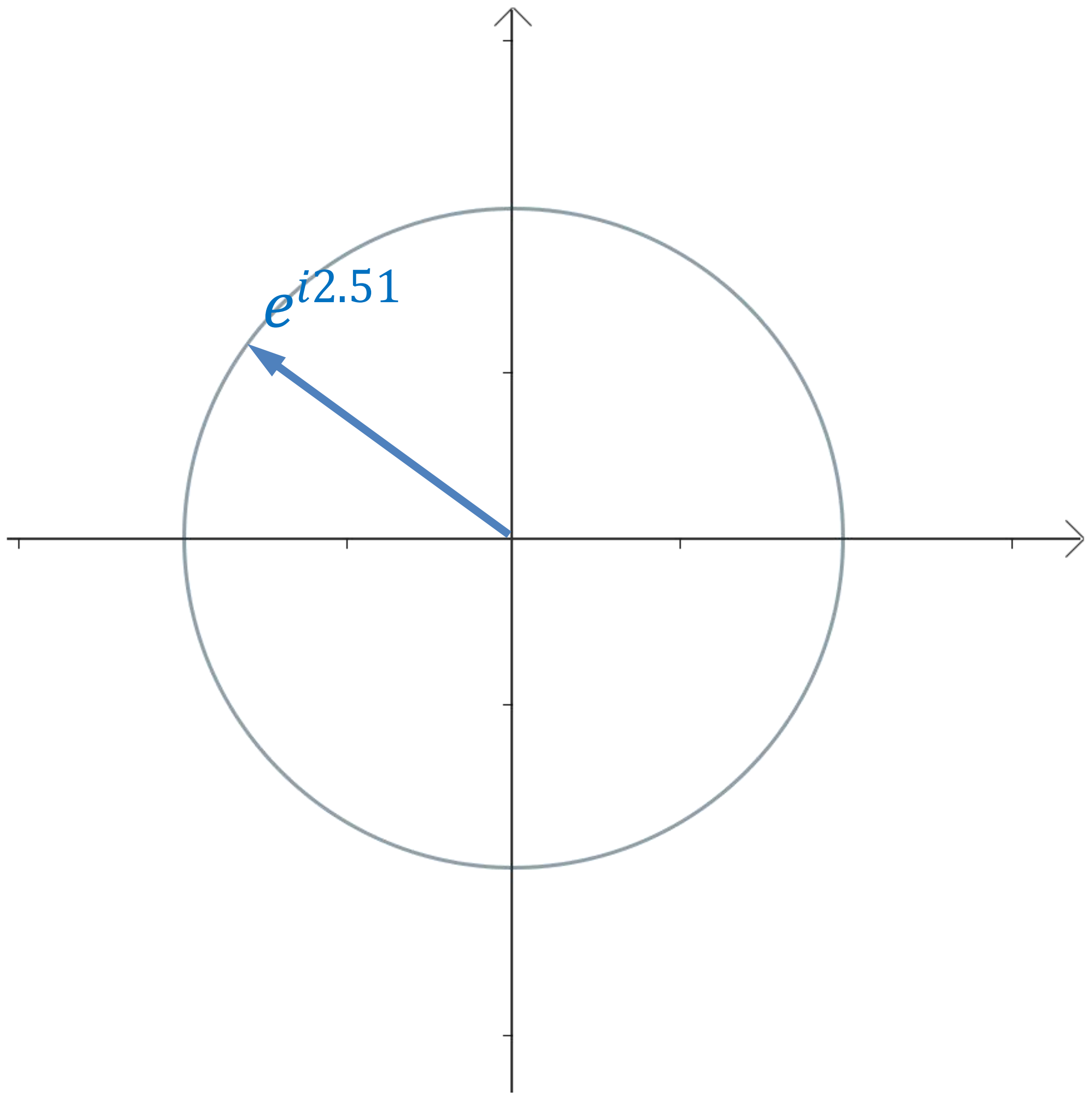
By adjusting the amplitudes of this bunch of periodic functions, plus the constant functions, the combination can be made into:

$$f(x) = c + \sum_{n=1}^{\infty} (a_n \sin\ (\frac{2\pi n}{T} x) + b_n \cos\ (\frac{2\pi n}{T} x)), c \in \backslash R$$

So the next question is how do you determine C,an, and Bn.

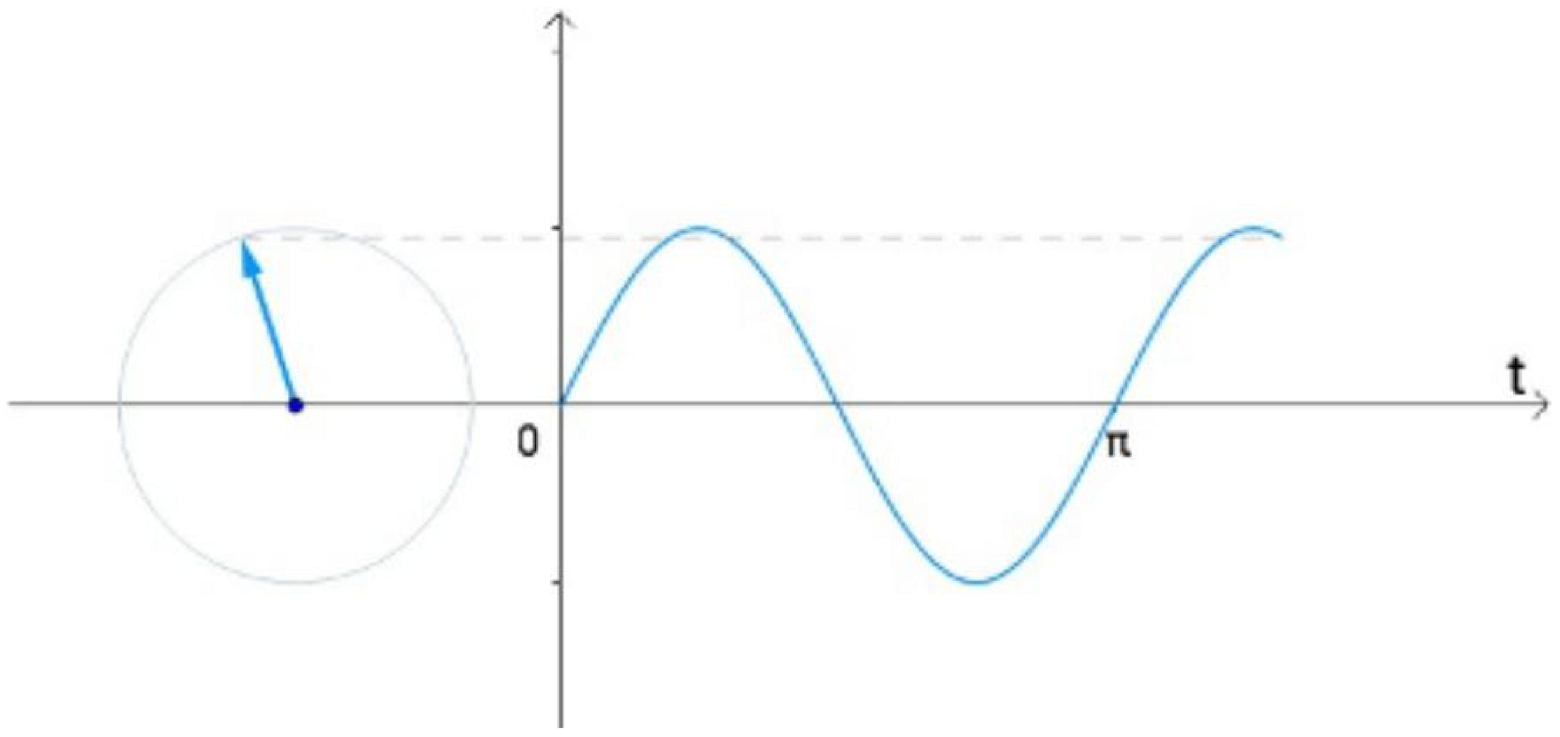
With complex numbers, we can express some things more concisely, especially when it comes to rotation operations.

Now, to do a little bit of an introduction, first of all, let's think about the complex plane.e^{in}

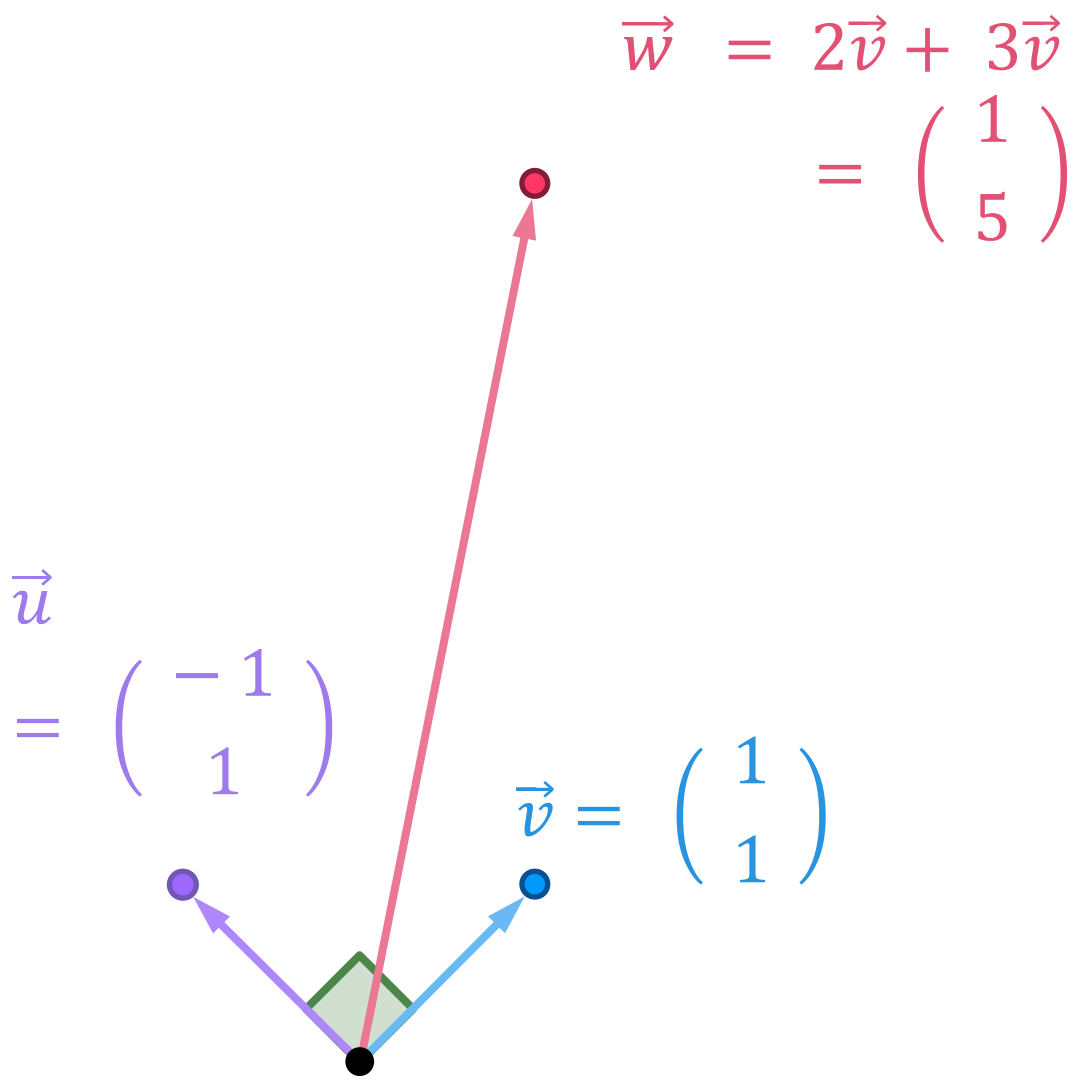


Naturally, t rotates once every 2 PI.

Add a w, that is, change the period of the rotation, and expand the imaginary part, which is the image of t ~ sin(wt).e^{iwt}



Generally speaking e^{iwt}, sin (WT) and COS (WT) can be regarded as vectors. How to find the orthogonal base coordinates?


$\vec{w} = 2\vec{v} + 3\vec{v} = \begin{pmatrix} 1 \\ 5 \end{pmatrix}$
$\vec{u} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$
$\vec{v} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

The coordinates of w under the basis are (2,3), that is to say, the vector f(x), f(x), is a linear combination of the following orthogonal basis:

$$\{1,\cos\left(\frac{2\pi n}{T}x\right),\sin\left(\frac{2\pi n}{T}x\right)\}$$

Can be obtained:

$$a_n=\frac{\int_0^T f(x)\cos(\frac{2\pi n}{T}x)\,dx}{\int_0^T cos^2\left(\frac{2\pi n}{T}x\right)dx}=\frac{2}{T}\int_0^T f(x)\cos\left(\frac{2\pi n}{T}x\right)dx$$

$$b_n=\frac{\int_0^T f(x)\sin(\frac{2\pi n}{T}x)\,dx}{\int_0^T sin^2\left(\frac{2\pi n}{T}x\right)dx}=\frac{2}{T}\int_0^T f(x)\sin\left(\frac{2\pi n}{T}x\right)dx$$

C can also be solved by dot product:

$$C=\frac{\int_0^T f(x)dx}{\int_0^T 1*1dx}=\frac{\int_0^T (x)dx}{T}$$

That is, C= A0/2, and the final result is:

$$f(x)=\frac{a_0}{2}+\sum_{n=1}^{\infty}(a_n\cos\left(\frac{2\pi nx}{T}\right)+b_n\sin\left(\frac{2\pi nx}{T}\right))$$

Among them

$$a_{n=}\frac{2}{T}\int_{x_0}^{x_0+T} fz(x)\cdot cos\left(\frac{2\pi nx}{T}\right)dx,n\in\{0\}\cup\mathbb{N}$$

$$b_n = \frac{2}{T}\int_{x_0}^{x_0+T} f(x)\cdot\sin\left(\frac{2\pi nx}{T}\right)dx, n\in\mathbb{N}$$

The Fourier series, above, can be written in complex form.

According to Euler's formula, it can be concluded that:

$$sin\theta = \frac{e^{i\theta}-e^{-i\theta}}{2i}$$

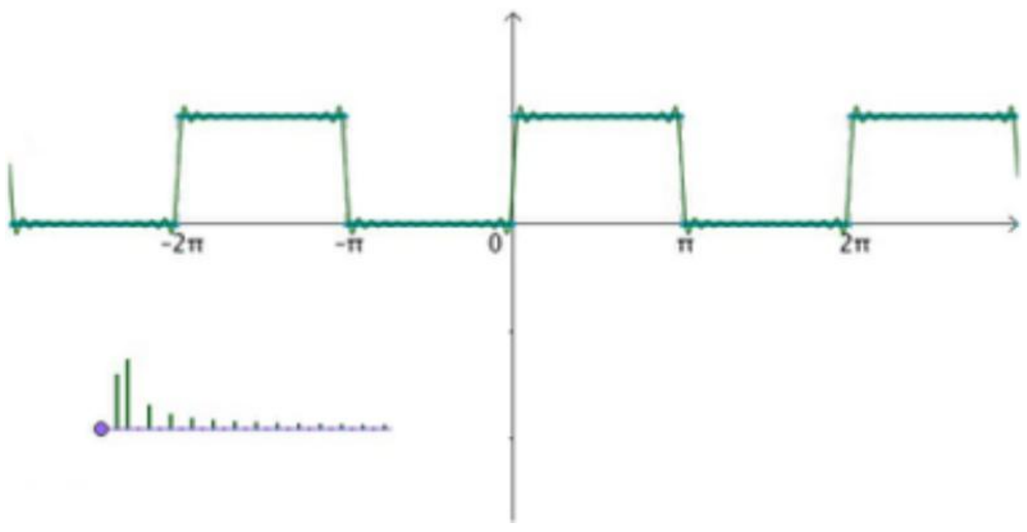
$$cos\theta = \frac{e^{i\theta}+e^{-i\theta}}{2}$$

For complex functions, the dot product defined is:

$$f(x)\cdot g(x) = \int_0^T \overline{f(x)}g(x)dx$$

The dot product is defined this way in order to guarantee $\vec{x}.\vec{x}\geq 0$

Fitting of Fourier series:



The larger n is, the higher frequency function is involved, and the higher the fitting degree is.

The height in the frequency domain represents the amplitude at this frequency, which is the coordinate component of this basis.

4.3 Encryption and decoding

4.3.1 Algorithm description

At a high level, GPV signature schemes are obtained by inserting lattice-based prototype-samplable functions into the classical "hash - symbol" paradigm, which dates back to the early public key cryptography [DH76, RSA78].

• Public and private keys are public tokens and trapdoors for M dimensional L, respectively.

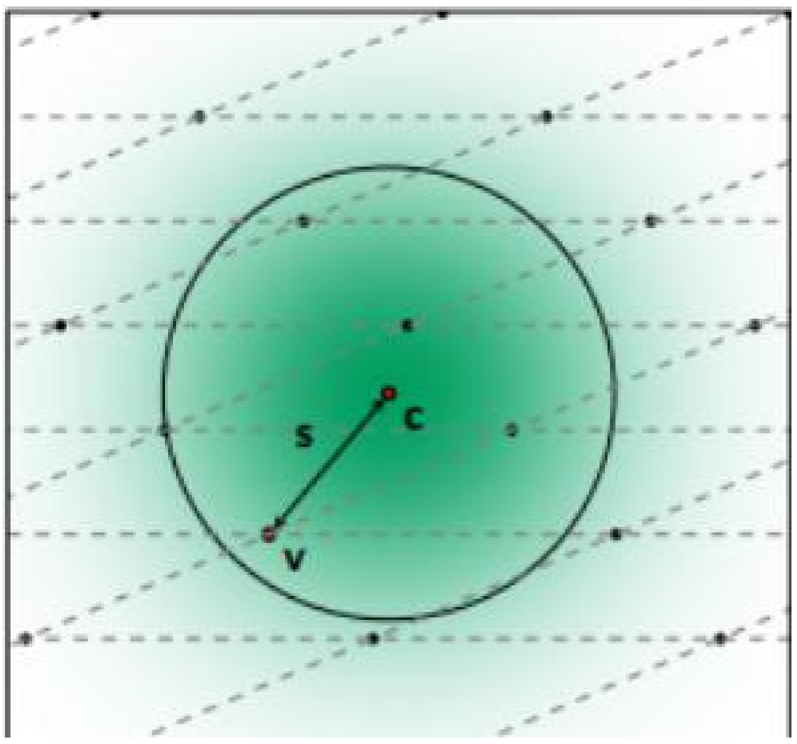
• Signature: Hash the message to the lattice companion set y+L using the public function, and then sample the signature x←Dy+L, s using the valve.S ≥ fileta (L) is a public parameter determined by the specific algorithm used to generate trapdoor and discrete Gaussian samples.Note that, according to the standard tail boundary, ≤s.$\|x\|\sqrt{m}$

• Verify a hypothetical signature x on a message: We hash the message to get the co-set y + L and simply check that x∈y + L and ≤s.$\|x\|\sqrt{m}$Q-ary SIS lattice L = L perp (A) and the equation fA(x) = Ax(A∈) are used as the pre-image sampling function of the bottom layer.$Z_q^{n* m}$

By modeling the common hash function as a "programmable" random Oracle, just like a typical hash and symbol paradigm.

When the key generator finds the average difficulty of a non-zero vector whose norm does not exceed 2s√m in the random lattice L(SIS whose concrete instance is the norm bounded by 2S √m), the scheme is unforgery under the selective message attack (Random-Oracle).

Examples of the construction and derivation of this problem are as follows:



We construct R = |x|/(+ 1) of a circle, and derive the key, signature and verification of the problem respectively. $Z_q x^n$

1) encryption

Keygen()

In R, the two matrices A and B meet the following conditions:

BA = 0;

B has a smaller coefficient;

Pk please A

Sk please B

2) signature

Sign(m,sk)

C is calculated by cA = H(m)

V ← D, d is the closest thing to C in the lattice A(B)

S please c - v

The signature sig is s=().s_1,s_2

3) verification

Verify(m,pk sig)

Iff

S is the shortest

sA = H(m)

4.3.2 Algorithm highlights

Security

A real Gaussian sampler is used internally, which ensures negligible leakage of information on the secret key, and almost an infinite number of signatures (over 2 ^ 64).

Compactness

Because of the use of NTRU latches, the Monster signature is much shorter than any lattice-based signature scheme with the same security guarantees, while the public key size is the same.

Speed

The use of fast Fourier sampling enables very fast implementation, with thousands of signatures per second on an ordinary computer. Verification is five to ten times faster.

Scalability

Operation degrees n cost O (nlogn), so you can use very long term security parameters at moderate cost.

RAM economy

Monster's enhanced key generation algorithm uses less than 30KB of RAM, 100 times more than previous designs such as NTRU Sign.Monster is compatible with small embedded devices with memory limitations.

4.3.3 Performance

Although quantum-resistant computers are the main driving force for the design and development of quantum-resistant signatures, the algorithm will be widely adopted only if it is also quite effective in today's world where there are no quantum-dependent computers. Monster, also developed as a robust signature, performs the following on a desktop computer (Intel®Core® I5-8259U on 2.3ghz, TurboBoost disabled) :

Monster-512	8.64	14336	5948.1	27933.0	897	666
Monster-1024	27.45	28672	2913.0	13650.0	1793	1280

The size (key generation RAM utilization, public key size, signature size) is expressed in bytes. The key generation time is measured in milliseconds. The private key size (not listed above) is about three times the size of the signature, and theoretically it could be compressed into a small PRNG seed (for example, 32 bytes) if the sigmer accepted to run the key generation algorithm each time.

For comparison, monster-512 is roughly equivalent to RSA-2048 from a classical security perspective, which uses 256 bytes for both signature and public key.OpenSSL's thoroughly optimized assembly implementation enables about 1,140 signatures per second on specific systems that take these steps. As a result, Monster's reference implementation is portable and does not use any inline assembly on x86 Cpus, is five times faster, and can be better scaled to larger sizes (to ensure long-term security).

4.3.4 Signature comparison

Lattice-based signature algorithm is considered as one of the most promising post-quantum cryptography algorithms due to its better balance in security, public-private key size and computing speed.Compared with the construction of cryptographic algorithms based on number theory, lattice-based algorithms can achieve significantly improved computing speed, higher security intensity and slightly increased communication overhead. Compared with other post-quantum cryptography methods, lattice-cipher has smaller public and private key sizes and better indicators such as security and computing speed. In addition, the lattice-based algorithm can realize encryption, digital signature, key exchange, attribute encryption, function encryption, homomorphic encryption and other existing cryptographic structures. The security of lattice-based algorithms depends on the difficulty of solving lattice problems. At the same (or even higher) level of security, the lattice-based algorithm has smaller public-private key sizes than the three constructs described above, computes faster, and can be used to construct multiple cryptographic primitives, making it more suitable for real-world applications.

Besides signature based on lattice, other resistance to the signature of the main types of code based on by quantum algorithm based on multivariate signature/based on the hash signatures such as classification, and through the screening of NIST quantum cryptography after selection, eventually to be included in the NIST the third round of the signature only Falcon (based on)/Rainbow (based on multivariate)/Crystal - Dilithium (based on).

	The public key size	Computing speed	Functional diversity	The number of drafts that belong to this class in the NIST standard solicitation
Case (Lattice)	small	fast	Very good	21 + 5 = 26
Code	smaller	faster	good	16 + 2 = 18
Multivariate	big	faster	good	2 + 7 = 9
Hash	big	faster	Co., LTD.	0 + 3 = 3

4.3.5 Safety analysis

The main source of security attack is the process of lattice reduction.

Key recovery

We first assume that a lattice is made of [q|h|0|1]. On this basis, we use lattice reduction to enumerate all lattice points of radius sphere · centered at the origin, and [g|f] can be obtained through high probability statistics.$\sqrt{2n}\sigma_{[f,g]}$Let the Gram-Schmidt norm of $(2n-B)$ be taken, which is approximately the norm of the shortest lattice vector generated by the orthogonal projection of the last B vector and the first $2n-B-1$

vector.λA screening algorithm performed on this projected lattice restores all vectors whose norm is less than a cube (see [Duc18]).$\sqrt{4/3}$λIf the projection of the key is in it, i.e., when ≤, we can recover the key vector from the projection of all the filtered high-probability vectors by using Babet's nearest neighbor plane algorithm.$\sqrt{B}\sigma_{[f,g]}\sqrt{4/3}$λThat's because all of the remaining gram-Schmidt norm is greater than, but much greater than.λλ$\sigma_{[f,g]}$

Suppose we can implement the Sieve algorithm in dimension B at the same cost as SVP Oracle in DBKZ algorithm, then it is easy to derive B and the value of the Gram-Schmidt norm given is correct only if the base lattice is first randomized.

Forge the signature

A forgery signature can be executed by finding a lattice of or distance between the same lattice as above. This task can also be solved by lattice reduction. One possibility is to use kannan embedding, which is to add (H(r| b| m), 0, K) to the lattice base, expanded by a row of zerps.Since the filtering algorithm generates many short vectors, we can certainly find such a vector (c,*,K) among them, and H(r||m)−c is a lattice point.

Set K≈, DBKZ algorithm [MW16, Corollary 2] gives the conditions for successful forgery:$\sqrt{q}$

$\frac{B}{2\pi c}^{n/B}\sqrt{q}$Beta or less.

However, since the factor is also present in beta, modulus Q actually has no effect on the optimal forgery attack.$\sqrt{q}$This is the best attack on instantiation. We follow the New Hope [ADPS16] approach (sometimes referred to as the "core SVP approach") to convert the block size B to specific bit security. According to [BDGL16, Laa16], this provides bit security.

Optimum attack cost

For Monster-512, we estimate that the complexity of the best attack is equal to the BKZ of block size B = 411.Latest method [19] [ADH + showed that n cost close to solve the problem of the shortest vector. Just take the first sieve asymptotic [BDGL16], will be 37, and the result is not considering the low dimensional of nearest neighbor search item. Each operation including at least one random access memory, the memory must contain 277 vector. Recent record [ADH + 19] in 112 dimensions were used in the circulation, the average each loop USES 16 above the door. Therefore, we believe that the minimum gate number 2120 + 19 + 4 = 2143 estimate is conservative.$\frac{n^3}{4B^2}B' \ = 3742^{19}\sqrt{1.5}^{112}$

Hybrid attack

Hybrid attack [How07] combines meet-in-the-middle algorithm and key recovery algorithm. Because it selects a sparse polynomial, it works well with NTRU. However, this is not the case here, so its effects are much smaller and balanced by the lack of siev-enumeration.

Algebraic attack

Although there is a wealth of algebraic structures in Monster, there is no known way to more than double the generic lattice equivalents of all the algorithms mentioned earlier.n^2

4.4 Consensus Mechanism

Because the traditional PoW and PoS consensus mechanism is difficult to balance efficiency and decentralization degree, the DMO adopts a new consensus mechanism, which we call PoT, i.e., Power of Trust, to effectively improve the TPS of the network.

4.4.1 PoT

4.4.1.1 Principle Description

The operation of PoT is divided into a series of epoches in terms of time, and each epoch is divided into multiple time slots. A block is generated in one slot, and the rights and interests are calculated according to the historical calculation before each epoch. At the beginning of each epoch, miners' nodes will calculate and generate an empty block meeting the current difficulty, and a group of participants

(P1, P2, P3...) will be derived from this empty block. , PT}. Then, each slot randomly selects N nodes from the Participants' set as basic interest representatives. These representatives use the set {S1, S2, S3, S4... SN} means, here N<T. Through the discrimination between the honest signature and the malicious signature in each slot, the dynamic credit granting is carried out.Here, based on logistic regression model, a dynamic measurement method of node trust is proposed, and its formula is:

$$trust_{cur}^{i} = \frac{1}{1 + e^{-a(\sum_{x=0}^{n-1} 9x - y * \sum_{x=0}^{n-1} x)}}$$

Where, trust(I) is the current trust given by the system to a node according to its behavior before the vote; N is the current NTH slot; Alpha is the slot number that the node accumulates to participate in. Theta x indicates whether node I normally participates in voting in the x slot, which is 1 normally, or 0 otherwise. Tx indicates whether node I votes maliciously in the x slot, with malicious 1 and vice versa. Gamma represents the penalty weight for malicious voting, which is set by the user. The greater the gamma value, the greater the penalty for malicious voting on the node.

The logistic regression model is quite fast in tracking the trust of a node, which is not conducive to reasonably judging the trust of a node. In order to avoid such a situation, this paper proposes an algorithm to correct the trust generated by the logistic regression model. According to the current trust of nodes and the trust in the last round of voting, the weight balance is carried out on the trust in the voting. The formula for measuring trust in the final vote is as follows:

$$trust(i)_{h}^{t} = \beta * trust_{cur}^{i} + (1 - \beta) * trust(i)_{h-1}^{t}$$

Where h represents the h slot within the current epoch. The trust degree of the node at the beginning of the first slot in the T-th epoch is equal to that at the end of the last slot in the T-th -1 epoch. In other words, the increase rate of the trust degree can be modified to make it not grow too fast, so as to avoid the centralization of the trust degree in the initial stage of the network.$\mathrm{trust}_{0}^{t} = \mathrm{trust}_{\mathrm{last}}^{t-1}\mathrm{trust}_{0}^{0} = \frac{1}{1+\mathrm{e}^{-\mathrm{a}(0-0)}} = 0.5$βInitially, the value is 1, because we don't know if the node is going to be evil at first.βThe change of is based on the cumulative deviation, and the specific change algorithm is determined by the following formula:$\xi_{h}^{t}\mathrm{trust}$

$$\beta = \mathrm{threshold} + c * \frac{\delta_{h}^{t}\mathrm{trust}^{i}}{1 + \xi_{h}^{t} trust}$$

Similarly, the accumulated trust bias at the beginning of the T-epoch was equal to the last participative consensus accumulated trust bias of the T-1 EPOCH, i.e., the parameter C was the user and was defined to control the weight of the response to the recent behavior of the node. $\xi_{h}^{t}\mathrm{trust} = \xi_{\mathrm{last}}^{t-1}\mathrm{trust}\xi_{h}^{t}\mathrm{trust} = 0$ Threshold is a threshold set with an initial value of 0 to prevent beta oversaturation from approaching 1. 25) In the sociological field, a person's credibility is better analyzed by historical trust than by current trust. Threshold is set to 0 to prevent oversaturation from reaching 1, but also to prevent the threshold from reaching 1

Does not rely too much on the direct evaluation of the current reliability function. $\mathrm{trust(i)}_{h}^{t}\delta_{h}^{t}\mathrm{trust}^{i}$ Represents the deviation of trust degree, and the calculation method is shown in the formula:

$$\delta_{h}^{t}\mathrm{trust}^{i} = |\mathrm{trust}_{h-1}^{t} - \mathrm{trust}_{\mathrm{cur}}^{i}|$$

At the T-epoch, the trust deviation of the node when it reached consensus on the h-th slot was equal to the difference between the absolute value of the current trust and the vote trust when the H-1st slot was voted. The final cumulative deviation non-formula is:

$$\xi_{h}^{t} trust = c * \delta_{h}^{t} trust + (1 - c) * \xi_{h-1}^{t} trust$$

As can be seen from the formula, the greater the value of C, the higher the weight of the recent trust bias given by the user is than the previous accumulated trust bias.

4.4.1.2 Node behavior discrimination strategy

The traditional consensus mechanism simply relies on raising the entry threshold, such as computing power or rights and interests, to maintain the stability and security of the blockchain network, but computing power and rights and interests based on currency age will make the blockchain network tend to be centralized.DMO puts forward a complete set of discrimination strategy for node behavior, through the detection and discrimination of node operation mechanism, to identify the malicious node, and carry out the punishment of trust.

The DMO defines node behavior as #good and #bad:

# good:

Each stakeholder's signature on a block is considered as a vote on the block. When voting, each stakeholder uses his or her private key to sign the block, which can be verified by using the public key. As long as the node participates in the signature of the generated block, meets the current difficulty, and is qualified to join the chain competition, its behavior is defined as #good;

# bad:

If the first N-1 stakeholder finds a problem with the block it wants to sign, it will not sign it according to the agreement. If the signature is still required, the NTH stakeholder with the right to package the block can decide whether or not to package the block before it is packaged. If it is not packaged, the block will be considered invalid, and all signature operations on the block will be deemed invalid, which will consume the trust of these stakeholders. Other nodes in the network verify the signature of the block if it is successfully packaged and competes with other blocks for trust. A block is considered illegal if some of the signatures in the block are illegal or if the block header hash is incorrect. All stakeholders who sign this block are considered to be malicious and use the corresponding punishment mechanism to punish trust.

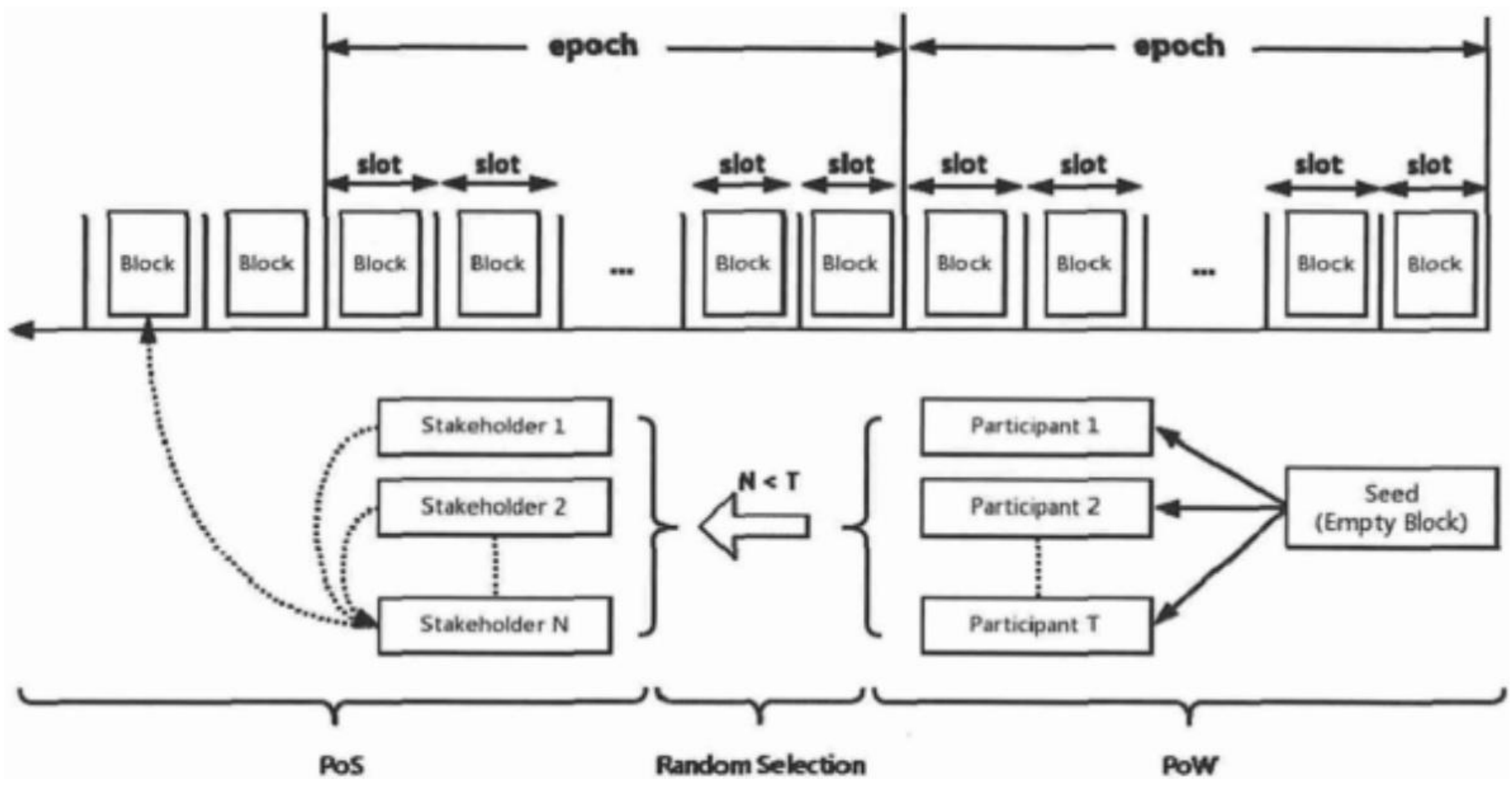
Based on the classic PBFT rule of 2/3, you can set a threshold for voting on the chain. Only in the case of more than 2/3 of the stakeholder votes in the block can the chain be qualified. The probability of a node becoming a stakeholder is calculated by the following formula:

$$\mathrm{prob}(s_i^{\exp}) = \frac{\sum_{t=1}^{n} \theta_t^i * \mu_t}{\sum_{t=1}^{n} \sigma_t}$$

Wherein, represents the number of successful participation of node I in the CONSENSUS within the EPOCH (T), represents the token rewards available within the EPOCH (T), represents all the token rewards issued within the EPOCH (t).$\theta_t^i \mu_t \sigma_t$

4.4.1.3 Block generation

In PoT, the block generation process is split in two: first, an empty block is generated by a simple amount of work; Then, through the stakeholder's vote to determine whether the block is qualified to chain. In the process of block generation, computing is introduced in order to avoid the equity crushing attack. In the process of putting transactions into block packaging and broadcasting to the network, the chain based equity proof mechanism is introduced. The block with high trust has a greater chance of chain compared with other blocks. Compared with the competition based solely on computing power, this method can greatly improve the network throughput and reduce the network resource consumption. Its specific architecture is shown in the figure below:



Among them, the stakeholder derives from PoT's coin following mechanism. After it comes into being, the blank block approved by it is signed with private key. The last stakeholder, namely, Stakeholder N, packages the transaction approved by it. After the block meeting the requirements is packaged, the chain can be connected. The PoT consensus mechanism produces the block as follows.

1) First, each miner attempts to generate a block header that contains the hash () of the previous block, the miner's address (), the block's height in the block chain, and a random value (nonce). $\mathrm{B_{prev}Adress_{miner}}$ Also, the block header cannot contain any transactions. Perform workload proof calculation. When the miners generate block head, if it meets the current network difficulty target, broadcast the block size to the whole network;

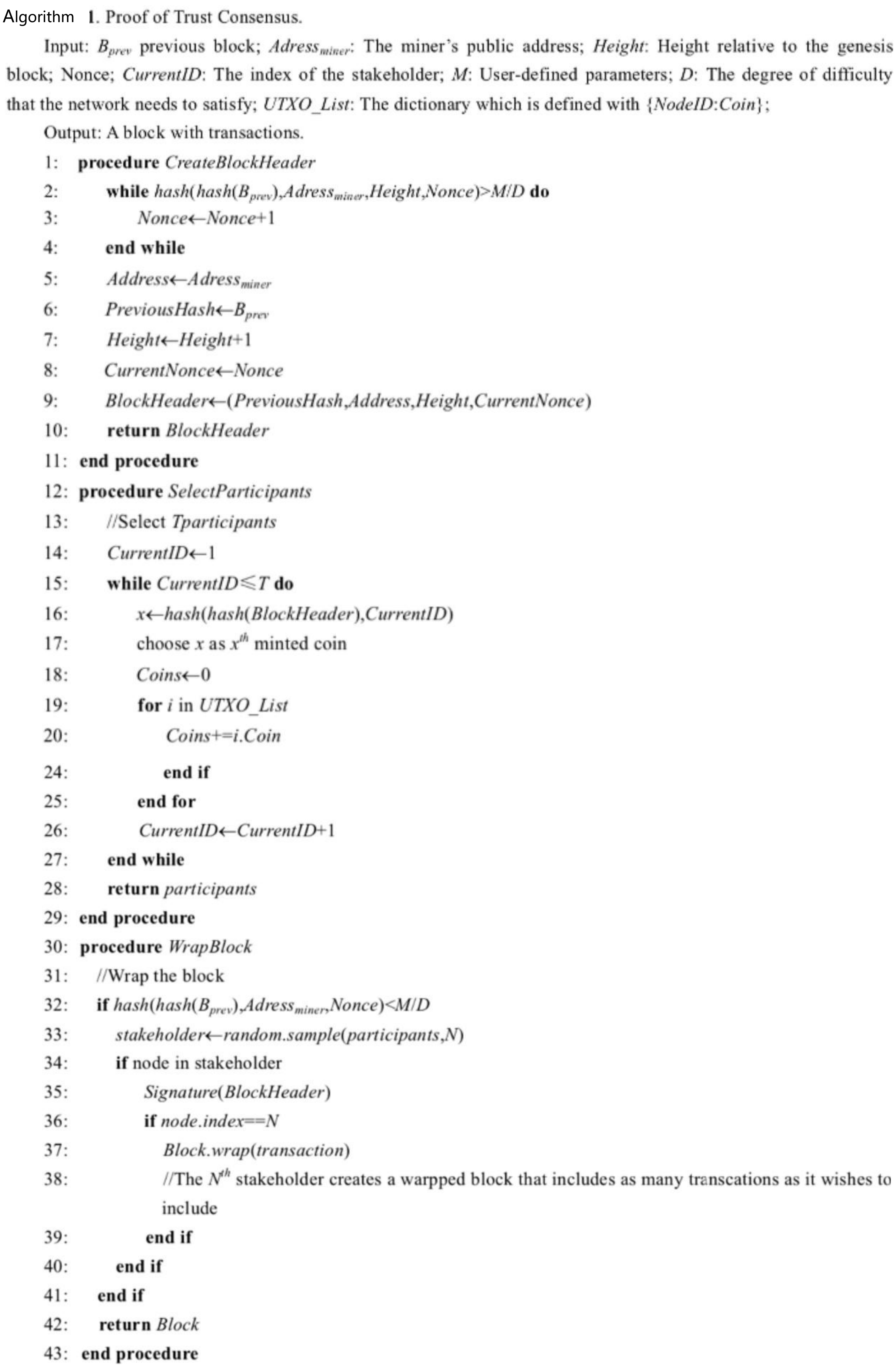
2) All nodes will take the hash value of the head of this block as the data of T participants.The specific algorithm is as follows: through the hash operation of T fixed values and the hash value of block head, the ith result x obtained will be used to determine the ith Participant. This process is to introduce a Coin following mechanism into PoT. Nodes intending to be participants sort all unspent outputs (UTXO) according to the dictionary. It is assumed that UTXO is not empty and the specific format is {NodeID, Coin}, where NodeID is the public key address of the node and Coin represents the number of coins held by the node. Also, CurrentID represents the ordinal number of the current Participant, and CurrentID ≤ T. The specific process is as follows: The node selects a random number X whose value is between 1 and the total number of UTXO in the system. In order to find the holder of the x coin, the node finds a minimum I, so that the number of UTXO tokens held by the node from the beginning of the list to I is not less than X. So the ith address is the owner of the XTH coin;

3) After Confirming T participants(participants).In each slot, N nodes are randomly selected as stakeholder participation consensus, where N<T. Each online rights representative checks whether the block head broadcast by the miner is valid. Once it is deemed valid, the rights representative checks whether it is one of the N rights representatives. If it is the first N-1 rights representative, it signs the block head with its private key. The process is to give stakeholder trust weight to the block and broadcast its signature. When the node confirms that it is the NTH equity representative, it packages as many transactions as possible into the block, adds the first N-1 signatures and its own signature to extend the block header, and finally generates the hash value of the block.

4) The NTH Stakeholder broadcasts the packaged block to the network. When other nodes receive the block and verify its validity, these nodes consider the block to be a legitimate extension of the block

chain.Considering that the network will expand greatly in the future, if two blocks are propagated across the whole network at the same time, then the two blocks will compete on the chain based proof of interest. When a block is extended to the chain, all stakeholder actions that sign the block are defined as #good;

5) After the block is generated, the incentives will be collected by the NTH Stakeholder and shared among the miners and the first N-1 stakeholder.The pseudo-code algorithm for consensus block generation is as follows :



Algorithm 1. Proof of Trust Consensus.

Input: B_{prev} previous block; $Adress_{miner}$: The miner's public address; *Height*: Height relative to the genesis block; Nonce; *CurrentID*: The index of the stakeholder; *M*: User-defined parameters; *D*: The degree of difficulty that the network needs to satisfy; *UTXO_List*: The dictionary which is defined with {*NodeID*:*Coin*};

Output: A block with transactions.

```
1:  procedure CreateBlockHeader
2:      while $hash(hash(B_{prev}),Adress_{miner},Height,Nonce)>M/D$ do
3:          $Nonce \leftarrow Nonce+1$
4:      end while
5:      $Address \leftarrow Adress_{miner}$
6:      $PreviousHash \leftarrow B_{prev}$
7:      $Height \leftarrow Height+1$
8:      $CurrentNonce \leftarrow Nonce$
9:      $BlockHeader \leftarrow (PreviousHash,Address,Height,CurrentNonce)$
10:     return BlockHeader
11: end procedure
12: procedure SelectParticipants
13:     //Select Tparticipants
14:     $CurrentID \leftarrow 1$
15:     while $CurrentID \leqslant T$ do
16:         $x \leftarrow hash(hash(BlockHeader),CurrentID)$
17:         choose $x$ as $x^{th}$ minted coin
18:         $Coins \leftarrow 0$
19:         for $i$ in UTXO_List
20:             $Coins+=i.Coin$
24:             end if
25:         end for
26:         $CurrentID \leftarrow CurrentID+1$
27:     end while
28:     return participants
29: end procedure
30: procedure WrapBlock
31:   //Wrap the block
32:   if $hash(hash(B_{prev}),Adress_{miner},Nonce)<M/D$
33:     $stakeholder \leftarrow random.sample(participants,N)$
34:     if node in stakeholder
35:         Signature(BlockHeader)
36:         if $node.index==N$
37:             Block.wrap(transaction)
38:             //The $N^{th}$ stakeholder creates a warpped block that includes as many transcations as it wishes to
                include
39:         end if
40:     end if
41:   end if
42:   return Block
43: end procedure
```

4.4.1.4 Credit consumption

Trust is not only a reward for nodes' honest participation in block generation, but also a manifestation of node reputation. To maintain the number of online nodes in the network, a credit consumption mechanism is introduced in PoT. At the same time, due to the characteristics of logistic regression model, the trust degree of nodes has a credit limit.

1) Credit consumption

Credit consumption is set to ensure the participation of nodes, rather than to allow nodes to participate in the generation of blocks in the way of consuming credit. Because when the network stakeholders are too few, the security and stability of the network cannot be guaranteed. In this mechanism, the credibility of nodes decreases over time. Even if the node is not a stakeholder, it must participate in the validation of the block in order to ensure network security. As long as it participates in the validation of the block, there will be no credit consumption. The specific consumption algorithm of node I trust is shown in the formula:

$$trust_{cur}^{i} = \begin{cases} \dfrac{1}{1 + e^{-a(\sum_{x=0}^{n-1} 9x - y * \sum_{x=0}^{n-1} \tau_x)}} \\ trust_{last}^{i} * e^{-D * \Delta B} \end{cases}$$

Δ B said block interval, the last time to participate in the generated block and the gap between the current block (start from 0), the algorithm is as follows: if just two blocks are continuous, then Δ B = 0.$\Delta\mathrm{B} = \mathrm{B}_{\mathrm{current}} - \mathrm{B}_{\mathrm{previous}}$

At this point, when nodes continue to participate in block generation with current trust, the trust consumption function will not be executed, which ensures that nodes in the network actively participate in block validation online to the greatest extent. D value represents the difficulty value of the network in the current period of time. The higher the value of D, the more repeated attempts are needed to find a valid block. The value of D is dynamically adjusted according to the outbound rate of the network, and the outbound rate of the network is finally maintained at a stable level. The increase in difficulty means that more trust weighting is needed on the block chain, so the chain loading cost will increase. Rational nodes may choose not to participate in block generation temporarily, in order to seek for the time when the network is less difficult and the opportunity is greater. When the node participation is too low, the probability of success of the malicious node is higher. Therefore, while increasing the difficulty of the network, it also increases the credit consumption of nodes that do not participate in the block generation, which will help to improve the participation of nodes and maintain the operation security of the block chain network.

2) The credit limit

The nodes in PoT can determine their voting weight without considering their own resources, and their upper limit of trust is 1, which solves the centralization problem caused by currency aging accumulation in the original PoS protocol to some extent. In the traditional PoS mechanism, the increase of voting weight is linear, while the increase of trust in PoT is amorphous. The increasing rate of trust will decrease over time and finally approach to 0, so the network centralization problem caused by excessive trust in a single node will not be caused.

4.4.1.5 Voting mechanism

In PoT, the original interest based on the age of the currency or the ownership of the cryptocurrency is changed to an interest based on the trust of the node itself, no longer dependent on the resource held by the node itself. Stakeholder confers trust degree on block through signature of block. The higher the trust degree is, the faster the hash result will be obtained. The calculation formula is as follows:

$$\mathrm{hash}(\mathrm{hash}(B_{Drev}),\mathrm{Block}_{cur},A,t) \le \delta_t * \mathrm{trust}(A) * M/D$$

Here, is the pre-block of the node voting block; B_{Drev} The address of the voting node is A, and its trust is; $\mathrm{trust}(A)\mathrm{Block}_{cur}$ It is the packaged block of Stakeholder N; T is the timestamp of the World Standard time (UTC); M∈[1, D] is a constant, which can change dynamically according to the real-time condition of the network; D represents difficulty; δ_t Is the time that has elapsed since it was created. B_{Drev} When a block is uploaded to the entire network, the initial probability of the block satisfying the formula is very low;But over time, the probability of success will gradually increase, and the first block to be calculated can become a legitimate extension of the block chain. For A stakeholder node with address A, its trust is locked, that is, the node is only allowed to vote once. Stakeholder ensures the effectiveness of their signature based on the provided address A and the timestamp T of the above formula.

4.4.2 Attack analysis

4.4.2.1 Bribery attack

The traditional double flower attack scenario is as follows:

1) The attacker initiates a transaction that is subsequently withdrawn by himself;

2) After the transaction, the attacker starts to establish a side chain on the block before the block of the exchange;(3) When a new transaction enters the block and sufficient confirmed numbers (6 blocks) are obtained, and the length of the attacker's side chain exceeds the main chain, the attacker's side chain becomes the main chain. The first transaction initiated by the attacker is judged to be invalid, and the double-flower attack is successful.

In order to successfully carry out a double-blossom attack, the attacker must control more than 50% of the network resources (PoW computing power, PoS equity) during the whole attack process. Controlling 51 percent of the scrip in circulation is very difficult compared to controlling 51 percent of the computing power. But that doesn't mean there's no way to attack in a proof of interest. An attacker can offer rewards to nodes that are willing to forge new blocks on the blocks he designates. If the attack fails, the user who participates in the attack does not suffer much. As long as the amount of the bribe is less than the price of the commodity, it is always profitable for the attacker.

In PoT, a bribe attack is not feasible because the stakeholder is different each time the block is generated and is uncertain before the block is generated. Once caught voting in bad faith, the penalty for trust is high, and only the NTH stakeholder has the power to package the deal. Unless the attacker finds the NTH stakeholder in advance, the stakeholder will be punished by the trust once it packages the invalid transaction.Because of the punishment coefficient, the high degree of trust punishment is unbearable for it.

The proof process is shown in the following formula:

$$trust^{i}_{9x} = \frac{1}{1 + e^{-a(\sum_{x=0}^{n-1} 9x)}}$$

$$trust^{i}_{\tau_x} = \frac{1}{1 + e^{-a(\sum_{x=0}^{n-1} 9x - y * \sum_{x=0}^{n-1} \tau_x)}}$$

$$\lim_{n\to\infty} Rat^{i}_{\tau_x,\vartheta_x} = \lim_{n\to\infty} \frac{1 + e^{-a(\sum_{x=0}^{n-1} \vartheta_x)}}{1 + e^{-a(\sum_{x=0}^{n-1} \vartheta_x - y * \sum_{x=0}^{n-1} \tau_x)}} = 0$$

The three formulas are as follows: trust obtained by node I honestly participating in the consensus;The updated trust degree of node I after the honest vote and the malicious vote is recognized by the system;The proportion of the trust of the malicious vote after the honest vote of Node I and the trust of the previous honest vote.
Assumptions:$9_x = \tau_x = a = 1$, $\gamma = 2$: $Rat^{i}_{\tau_x,\vartheta_x} = \frac{1+e^{-a*\vartheta_x}}{1+e^{-a(\vartheta_x - \gamma*\tau_x)}} = \frac{1+e^{-a}}{1+e^{a\gamma - a}} = 0.36$

In the second slot of the epoch, after an honest node had participated in the consensus once, the system identified malicious attackers with only 36% of the trust after a malicious vote.And the trust penalty increases rapidly with the number of rounds and time, with the greater the alpha, the heavier the penalty. The existence of such a degree of punishment is unacceptable to any rational node.

4.4.2.2 Cumulative attack

The original version of the cumulative attack is the age cumulative attack, which is launched against PPCoin and other systems that use age of coins as users' rights and interests. In later versions of PPCoin, UTXOs was limited to 90 days. In Novacoin and BlackCoin, there are also limits on coin age. However, by limiting the age of the currency, the likelihood of attack is greatly reduced and the benefit of using age as an equity is reduced. In PoT, the accumulation of trust is not linear, but based on a nonlinear logistic regression model.

In PoT, if you want a cumulative attack, you need to accumulate trust. DMO network modified logistic algorithm makes the accumulation of trust is very long, malicious behavior, once found, and the node to node trust penalties will be any rational nodes are unsustainable, therefore in the PoT, want to attack through the accumulated trust income is not enough to offset the loss of trust to punish its. If the attacker wants to bypass the trust, only through their own interests to generate blocks and other blocks on the chain competition, the probability of success is also quite low.

Proof: First, the attacker wants to control the stakeholder to vote 2/3. In a network with N stakeholder nodes, the attacker must control at least 2/3 of the stakeholder nodes.At the same time, the attacker also needs the last stakeholder to package his fake transaction into the block, that is, one of the stakeholders controlled by the attacker must be the deal packager. Therefore, the total probability of the attacker's success is:

$$Suc_n^i = p * \sum_{k=\left\lceil\frac{2n}{3}\right\rceil-1}^{n-1} c_{n-1}^k \, p^k (1-p)^{n-1-k}$$

P is the probability of becoming a stakeholder. Suppose a network of 6 stakeholders holds 20% of the total equity, then the attacker bypasses the trust and succeeds in the online competition. Then the probability of his success is:

$$Suc_n^i = 0.2 * (C_5^3 * 0.2^3 * 0.8^2 + C_5^4 * 0.2^4 * 0.8^1 + C_5^5 * 0.2^5) = 0.011$$

Compared with its 20% interest, the 1% success rate is paltry.

4.4.2.3 Rights crushing attack

Interest crushing attack is a specific attack mode against PoS. The specific scenario is as follows: in the network running PoS protocol, if a node holds a low interest (such as 1%), the probability of its successful block generation is also 1%. Any rational node is willing to try forking, because in the pure PoS protocol, forking does not need to consume any resources, and the only consumption is the currency age. If the block is not accepted, the coin age will not be consumed. Although this reduces the value of cryptocurrency across the network, the nodes don't care because they have so little interest in it. In PoT, only N randomly generated stakeholders have the right to extend the block chain, while miners are only responsible for generating blocks for the stakeholder to sign, without the right to decide. It is very clear when making decision, won't have ambiguities, have very strong management ability. Under normal circumstances, it will not experience any fork, because N stakeholders are cooperative production blocks rather than competition.

5 The DMO builds a community autonomous data structure

DMO is a decentralized public chain with quantum resistance, which can be audited, publicly verified and designed according to the incentive mode. Miners get paid by holding and participating in ecological construction. Miners are paid only when the algorithm is activated and their services are properly provided according to the anti-Matthew algorithm and Fourier algorithm.

5.1 Application Basis

DMO is an independent data structure that is autonomous by the community and adopts extensible anti-quantum signature algorithm. In order to ensure its independent operation, the DMO contract platform has the following preliminary application basis:

Expansion mechanism

Based on the segment tag built in DAPP, the response mechanism of DMO divides all DMOS into Anti Matthew mining DMO and responsive mining DMO, in which anti Matthew mining is judged and transformed every 24h.

Contracts for the game

The first ecological application contract launched by DMO is a crowdfunding game in which contracts are linked. Each round of crowdfunding has a specified goal and time. After reaching the goal within the specified time, the next round of crowdfunding will be carried out, and the crowdfunding goal will be enlarged. Players who participate in the game will get the results in a few rounds. When crowdfunding ends, the last participating address will receive a bonus from the grand prize pool.

The insurance agreement

When the round crowdfunding of DMO contract game is terminated, the DMO participated by the user will be automatically recorded by the contract and the insurance agreement will be triggered at the same time. The DMO terminated by crowdfunding in the insurance agreement will receive double insurance compensation within a certain period of time. The proportion of insurance mining is determined by the liquidity of the flowing mine pool, and the compensation range is 1 ‰ to 1%.

Cross-chain contract trading

Cross-chain contract trading is a kind of agreed-upon trading to solve THE liquidity of DMO. It is different from deep-based matchmaking and point-to-point trading based on price discovery. Instead, it makes a mining pool based on DMO to store the chips of both parties, and completes the cross-chain trading in the form of order book. Thus, when a cross-chain transaction is completed, a transfer is actually made within the agreement at a price allowed by both parties (using the AMM mechanism).

5.2 Mining period

We allocate the unique licensed DMO under Digital Monster Operation's network: the total circulation of DMO is 1 billion.
Of these, 3.5 million were used for the original issue,
200 million pieces are used for Fourier algorithm response mining,
100 million for insurance agreement mining
696.5 million pieces for anti-Matthew mining.
Among them, the anti-Matthew algorithm mining will complete all DMO mining in nine stages in the next ten years. 696500000 DMOS were excavated by anti-Matthew algorithm, and the mining rate decreased in different stages. The mining period is shown in the table below:

2021.1-2021.06	2021.07-2021.12	2022.01-2022.06	2022.07-2022.12	2023.01-2023.06	2023.07-2024.06	2024.07-2025.12	2026.01-2027.12	2028.01-2030.12
20%/ month	15% month	10%/ month	8%/ month	6%/ month	4%/ month	3%/ month	2.5%/ month	1.2%/ month

6 Ecology

6.1Black mirror

Social software based on blockchain technology should essentially explore the free and happy parts of human nature rather than the user scenarios that stick to traditional social, just as the business solutions of traditional finance do not apply to DEFI (decentralized finance).

Black Mirror is a new generation of blockchain social product that is different from traditional social products from technical architecture to design concept. It adopts point-to-point technology and is aimed at minimalists.

Core functions:

Face recognition: the user through the facial recognition technology rapid login, and share content (image/video) and I have relevance, namely the user only upload content that is related to oneself, simplifying the social redundant information, strengthen the direct social people, this is the embodiment of "less is more" intuitive, more effectively put an end to fraud, such as good money after bad money behavior way, but the user can choose to hide personal information, V mask.

Social computing power: The value of social networking itself is visualized through computing power. Adding friends with greater social influence can help you improve your computing power. Social experts can occupy a larger proportion of computing power in the network, but they need to maintain the network carefully, so as to help the whole network achieve dynamic balance.

Value prioritization: Different from the previous information flow processing mode in reverse chronological order, Black Mirror adopts the information flow pushing scheme in reverse chronological order to prioritize its own more valuable information. As the pace of modern society accelerates, many people have to prioritize more important things to focus on, such as family and friends. For minimalists, prioritizing valuable information comes first.

Black Mirror will issue a contract based on the DMO, which is an interesting attempt by the DMO to explore a new social approach to blockchain.

6.2 Gamero

Gamero is a new game distribution strategy in which game publishers' game revenue is distributed directly to anonymous accounts and can be returned to users and advertising platforms using standardized rebates, including, but not limited to, payments based on effective downloads and effective browsing.

Oracle technology is used to complete the collection of external data and integrate the global computing power for distribution.

6.3 Dopay

Dopay provides a way to decentralize cross-chain clearing, which works in a similar way to cross-chain contract trading but tends to pool in greater depth and capacity, while serving traditional enterprises' on-chain economies, including the adoption of other currencies that are already accepted and well known.

Dopay contracts are a new way of managing blockchain assets as an enterprise. Dopay contracts are the equivalent of aggresively applying multiple contracts, such as clearing/transfer/asset management/payment interface, into a set of visual management tools. A few Dmos can be pledged to use this public service.

Reference

{1}Thomas Pornin and Thomas Prest

More Efficient Algorithms for the NTRU Key Generation using the Field Norm

{2}Xingye Lu, Man Ho Au and Zhenfei Zhang

Raptor:A Practical Lattice-Based(Linkable) Ring Signature

{3}Thomas Pornin

New Efficient, Constant-Time Implementations of Falcon

{4}Pierre-Alain Fouque,Paul Kirchner,Mehdi Tibouchi,Alexandre Wallet and Yang Yu

Key Recovery from Gram-Schmidt Norm Leakage in Hash-and-Sign Signatures over NTRU Lattices

{5}James Howe,Thomas Prest,Thomas Ricosset and Melissa Rossi Isochronous

Gaussian Sampling: From Inception to Implementation

{6}Nakamoto S.

Bitcoin:A peer-to-peer electronic cash system

{7}Buterin V.

A next-generation smart contract and decentralized application platform.

{8}King S,Nadal S.

Ppcoin:Peer-to-peer crypto-currency with proof-of-stake.

{9}Poon J, Dryja T.

The bitcoin lightning network: Scalable off-chain instant payments.

{10}Cachin C.

Architecture of the hyperledgerblockchain fabric.In: Procf of the Workshop on Distributed Cryptocurrencies and Consensus Ledgers.

{11}Yuan Y, Wang FY.

Blockchain: The state of the art and future trends. Acta Automatica Sinica,