SPHINX

An open-source post-quantum

Blockchain layer 1

https://linktr.ee/sphinx.org

06/08/2023

Abstract;

The rise of Quantum-computing urging us to develop and implement quantum-resistant cryptographic algorithms that could withstand the computational might of these emerging technologies. This is a call to action for global community, we initiated to lead an open-source post-quantum blockchain layer 1 project, its open to anyone who want to join and contribute.

TABLE OF CONTENTS

INTRODUCTION

Blockchain technology has transformed industries by providing decentralized and immutable transactional systems. However, the emergence of quantum-computers poses a significant threat to the security foundations of existing blockchain networks. Quantum-computers have the potential ability to break traditional cryptographic algorithms, compromising the integrity and confidentiality of blockchain transactions.

To mitigate this imminent threat, it is crucial to develop a new design of new layer solution that can withstand attacks from quantum-computers. Post-quantum cryptography offers a promising avenue for addressing these vulnerabilities. By harnessing the principles of quantum-resistant algorithms, we can ensure the long-term security and sustainability of blockchain technology.

The goal of our blockchain project is to create a robust and secure blockchain infrastructure that enabled post-quantum cryptography, we aim to develop a system that not only achieves quantum resistance but also provides enhanced scalability and improved performance, laying the foundation for the future of blockchain technology.

In this white paper, we present a comprehensive overview of our project, highlighting the unique value proposition it offers and detailing its key components and features. By combining post-quantum cryptographic algorithms, advanced consensus mechanisms, and innovative architectural design principles, we aim to revolutionize the blockchain industry and establish a quantum-resistant future.

PROBLEM STATEMENT

Classical-century 1.1

The concept of blockchain was first proposed in 1991 by Stuart Haber and W. Scott Stornetta. However, it wasn't until 2008 that blockchain technology was first used in practice, with the launch of Bitcoin as a peer-to-peer electronic cash system that uses blockchain technology to track transactions.

Blockchain works by creating a chain of blocks, each of which contains a set of transactions. Each block is linked to the previous block using cryptography, which makes it very difficult to tamper with the data in the blockchain. When a new transaction is submitted to the network, it is added to the next block in the chain. Once a block is added to the chain, it is very difficult to remove or modify, as this would require changing all of the subsequent blocks in the chain.

Fast forward 15 years ago in this classical-century, Bitcoin's resilience still stands strong, this age has been a testament to the rightness of its design and the efficacy of its components.

From bitcoin’s Founder (Satoshi Nakamoto) we all here celebrate the enduring legacy of Bitcoin, a pioneer that has left an indelible mark on the world, he/ she taughts us all here a lesson how to build a tamper-proof systems in advancement in the digital age, especially in the cyber-security defense principle.

Classical-century 1.2

Satoshi Nakamoto was given us a lessons what does component to used to build a tamper-proof peer-to-peer systems, since that its inspired global community project to following the design component of bitcoin principle.

Overview the important blockchain component;

1. SECP256K1 for generate key pair
   The secp256k1 curve is defined by the equation: $y^2 = x^3 + 7 \pmod p$, where p is a large prime number that defines the finite field, and x and y are coordinates on the curve.

2. ECDSA for digital signature
   The ECDSA (Elliptic Curve Digital Signature Algorithm) is based on the properties of the secp256k1 curve, which is defined by the equation: $y^2 \equiv x^3 + 7 \pmod p$, where p is a large prime number that defines the finite field, x and y are coordinates on the curve.

3. SHA2-256 for hash function
   The underlying mathematical operations in SHA-256 can be quite complex, but we can simplify them to give high-level understanding:

i. ***Ch (E, F, G)*:**
- Calculate (*E* and *F*) AND (NOT *E* and *G*).
- Perform an XOR operation between the two results.

ii. ***Ma(A, B, C)*:**
- Calculate (*A* AND *B*) XOR (*A* AND *C*) XOR (*B* AND *C*).

iii.***Σ0(A)*:**

- Perform bitwise rotations of *A* by *2, 13,* and 22 bits to the right.
- Perform three XOR operations between the rotated results.

iv.***Σ1(E)*:**

- Perform bitwise rotations of *E* by *6, 11,* and 25 bits to the right.
- Perform three XOR operations between the
- rotated results.

Modulo *2^32* for SHA-256 or *2^64* for SHA-512. This means that the resulting values are truncated to fit within a 32-bit or 64-bit space, respectively.

Quantum-century 1.3

For 15 years in the classical-century blockchain is proven can not be broken, but it would different in the quantum century. In principle there are no single wrong in the design principle of blockchain technology but it would different in terms of component to used.

Today in 2023 we talking about quantum-computers is not sounds like we talk about this mechine back to in 2001's, since thats century quantum-computers has been significant grown and promising to solve the largest problem in our today's technology limitations.

Quantum-computing landscape:
IBM Roadmap.

IBM quantum centric 100,000 qubits.

(EuroHPC JU) quantum-computers.

The end of Classical-computers;

For decades Moore's Law has guided the exponential growth of classical computing power. It states that the number of transistors on a microchip doubles approximately every two years, leading to remarkable advancements. However, we are approaching the maximum capacity of classical computing using this law.

As transistors become smaller, it face significant challenges in miniaturization. The physical limitations of classical computers would hinder further progress, a limitation signals to the end of Classical-algorithm, prompting industry to found new approaches and it might by eliminated transistors the quantum-computing is coming as the solution.

ECDSA and SECP256K1

ECDLP can be broken by large scale quantum-computers, the best known vulnerability of ECDLP is Shor's algorithm with Polynomial-runtime capable to factoring ECDLP into their prime factors, this is a critical step in breaking the security of Elliptic curve.

Shor's presented two polynomial time quantum algorithms, one for factoring integers, the other for computing discrete logarithms in finite fields.

We are given an instance of the ECDLP as described above;

> Let $P \in E(Fp)$ be a fixed generator of a cyclic subgroup of $E(Fp)$ of known order $ord(P) = r$, let $Q \in hPi$ be a fixed element in the subgroup generated by P; our goal is to find the unique integer $m \in \{1, \ldots, r\}$ such that $Q = [m]P$.

With this step formula the ECDLP or Large integers that safeguarding the signature and key could be broken by large-scale quantum-computers.

However something like this would never happened in over night, quantum-computers needed millions or billions of "Logical-qubits" to perform Shor's algorithm and breaking Elliptic curve with 256-bit.

But important to note that like or not we are all would come to the quantum-computing era, and its closer than we are realized. NSA as the large cyber security agency was warn for global community that we have deadline until 2035 to migrate into post-quantum cryptography, this is not only threat for blockchain but for entire digital infrastructure.

SHA Family hash function

In principle all of SHA family is designed by NSA, SHA2-256 is created in nearly 1990-2000 and then published in 2001, the design is using **Merkle–Damgård** construction.

NSA as the origin creator of SHA family was warn that SHA with 256 digests size would no longer safe in quantum-computers era, in this era that promising secured is SHA3 with 384 digests size.

Today the best theoretical known how to attack on hash function is using Grover's algorithm. To attack hash functions (SHA2-256) to finding private key, it can use a technique called parallel Grover's search.

In parallel Grover's search, we create a quantum oracle that maps the hashes of the private key to the private key itself. This can be done by using a quantum computer to simulate the hash functions;

> The time it takes to find the private key using parallel Grover's search is $O(\sqrt{N/2})$, where N is the number of possible inputs to the two hash functions. This is a quadratic speedup over the best known classical algorithm, which takes $O(N)$ time.

With this well known vulnerability we can not longer depend on this type of hash function, it might for **Keccak** or **Merkle–Damgård** construction would about double size into minimum 384 digest size.

SOLUTION STATEMENT

Post-quantum 1.1

After six years of research and development, the National Institute of Standards and Technology (NIST) has announced in 2022 the first four quantum-resistant encryption algorithms. The chosen algorithms Crystals-kyber for KEM's and SPHINCS+ for alternative digital signature, both them are designed to withstand attacks from future quantum computers, which are expected to be significantly more powerful than the current generation of machines.

The announcement marks a significant milestone in NIST's post-quantum cryptography standardization project. The chosen algorithms would now undergo further testing and evaluation before being finalized and published as NIST standards. Once finalized, these algorithms would be used to protect sensitive data from attack by quantum computers.

The development of quantum-resistant encryption is a critical step in ensuring the security of our digital infrastructure. As quantum computers become more powerful, they pose a serious threat to the security of our current encryption systems. The chosen algorithms would help to ensure that our data remains secure even in the face of this threat.

The significance of this announcement as a crucial step in safeguarding our cyber-security, sensitive data against potential cyberattacks from quantum-computers threat. NIST's exceptional expertise and dedication to pioneering technology, enabling us to proactively protect electronic information.

Post-quantum 1.2

Our decision to utilized Post-quantum cryptography is critical steps in this century for blockchain ecosystems, we can not again rely on single problem to safeguarding our network.

Key features that enable in our ecosystems;

1. Post-quantum Hybrid key
   We using curveX448 and Crystals-kyber 1024 and then we “merged” them into single key, the idea of merged is to combine their strengthness and then we called it with “SPHINXKey”.

2. Post-Quantum “Stateless” Digital signature
   We utilized post-quantum digital signature SPHINCS+, this signature is more larger and slower than another post-quantum digital signature scheme, but remember in the quantum-computers era the larger key size is more better, so we choosen for long terms security guarante.

3. SWIFFTX lattice-based Hash function
   SHA3 candidate in 2008’s but this construction is based on combination between SWIFFT and Cyclic/ ideal lattice construction. Lattice-based is very well known today that is resistant against quantum-computers, so we utilized 256 digest size of SWIFFTX called “SPHINXHash”.

4. ZK-STARK protocol

   To provide privacy guarantee, that is based on Zero-knowledge proof. ZK-STARK enables users to demonstrate knowledge of the private key and the ability to sign transactions correctly without actually disclosing the private key itself.

5. Proof-of-Work with a Distinctive Approach

   Our innovative consensus mechanism ensures randomness and significantly reduces energy consumption, fostering a sustainable and eco-friendly blockchain ecosystem.

6. Seamless Inter-chain Interaction

   Embracing cross-chain, bridge, atomic swap, and side-chain technologies, our network enables smooth communication and collaboration between diverse blockchain systems.

7. Scalability through Horizontal Sharding

   Leveraging horizontal-shard architecture, we achieve unprecedented scalability and seamless transaction processing, empowering a high-performance blockchain infrastructure.

8. Multilingual Smart Contracts

   Embrace creativity and flexibility with our platform's support for multiple programming languages, empowering users to create smart contracts with ease and in their preferred language.

Post-quantum Hybrid key

Overview the important part of post-quantum blockchain layer 1 project, its would about Encryption systems, Digital signature, and Hash function.

Crystals-Kyber

The underlying of crystals-kyber is based on MLWE + Lattice, the mathematical security reduction of the ring-LWE problem to MLWE that make it believed to be NP hard to solve against classical and quantum-computers, we used crystals-kyber 1024 that is equal to AES 256, means it would given us 256-bit security level.

1. Key Generation:

- Private Key: The private key consists of two small polynomials:
- Private Key (s) = $(-x^3 - x^2 + x, -x^3 - x)$
- Public Key: The public key consists of a matrix of random polynomials A and a vector of polynomials t:
    - Public Key (A, t) = (Matrix A, Vector t)

2. Encryption:

- To encrypt a message, convert it into a polynomial with binary coefficients. For example, if the message is 11 (1011 in binary):
- Encoded Message m_b = $x^3 + x + 1$
- Scale the message polynomial by multiplying it with the integer closest to q/2 (where q is a parameter).
- Encrypted Message m = 9 * m_b

3. Encryption Procedure:

- Calculate two values, u and v:
- $u = A^T * r + e_1$
- $v = t^T * r + e_2 + m$
- The ciphertext consists of these values: (ciphertext) = (u, v)

4. Decryption:

- Given private key s and ciphertext (u, v), compute a noisy result m_n:
  $m_n = v - s^T * u$

5. Message Recovery:

- Recover the original scaled message m_b by analyzing the coefficients of m_n:
  If a coefficient is closer to q/2, round to 1; if closer to 0, round to 0.

6. Final Step:

- Scale down the recovered polynomial m_b by dividing by 9:
- Decoded Message $m_b = (1/9) * m_b$
- Extract the bits of the polynomial to get the original message: 11

This process involves private and public keys for encryption and decryption. The encryption process involves polynomial operations and scaling. The decryption process reverses the encryption to recover the original message.

Elliptic Curve X488

While for CurveX488 is based on ECDLP that we known it would vulnerable attack by quantum-computers, we used curveX488 and it would given us 224-bit security level. However to break elliptic curve with 224-bit it would needed millions of logical qubits for quantum-computers.

1. Golden-Ratio Prime Base:

Hamburg selected a special prime number called a "Solinas trinomial prime" with a value of p = 2^448 - 2^224 - 1. He named it a "Goldilocks" prime because its structure is related to the golden ratio, which is a famous mathematical constant denoted by φ. This prime has a property that is useful for fast multiplication.

2. Curve Equation:

The mathematical equation used to define the curve is: $y^2 + x^2 = 1 - 39081 * x^2 * y^2$. This equation describes a curve in a specific shape. The number 39081 is used in a way that makes the curve's properties work well for cryptographic purposes.

3. Special Constant:

The value d = -39081 was chosen carefully. It's the smallest value with specific mathematical characteristics required for the curve's security. This choice of d is important to ensure that the curve remains secure.

4. Implementation Considerations:

Curve448 was designed to avoid common problems that can occur during its implementation in cryptographic systems. This careful design helps prevent potential vulnerabilities and ensures that the curve functions as intended in various scenarios.

Hybrid-key PQC encryption

If we combine both Crystals-kyber 1024 and curve X488 it would given us 480-bit security level, it would more computational resources requirement to breaking the encryption systems.

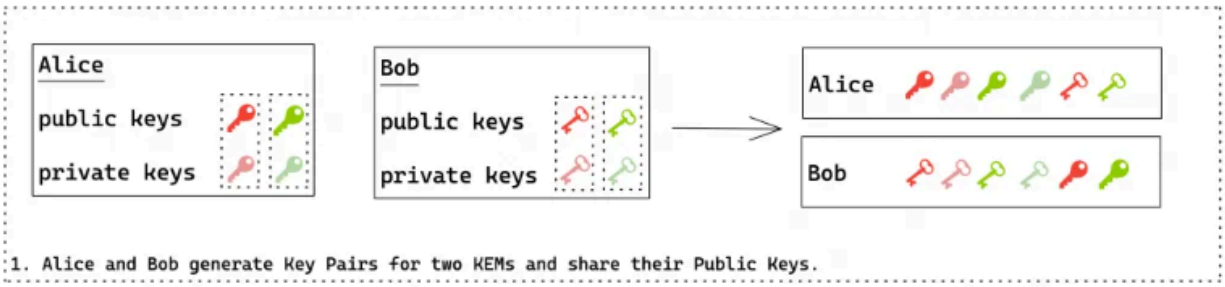
Kyber1024 and CurveX488

Method: Kyber1024-X448
Public Key (pk) =
ACDC644291611A014B41408A16D52479B513FBE15906B259660BB0579B1D9508 (first 32 bytes)
Private key (sk) =
690133C140927B6212D539458DB18DC0F28B7CEB549BF32C2FB03229A109C705 (first 32 bytes)
Cipher text (ct) =
488232A8878C312FE5988DD19EA04CBC892F25036569423E83061CAEF2C32FC7 (first 32 bytes)

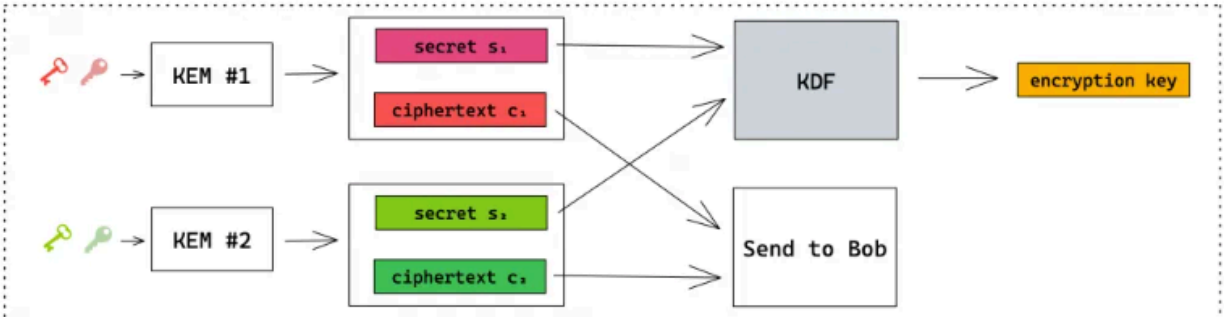
Shared key (Bob):
FFE3DA3744D79EEBAC20C6224B9EAE6B9622F5B8202C4227A7500984DC0943842
A1EE9A775E5CD699CAE5A3134925F666F0F1C78B9B3237F4EFA50EE7632F47485A9D4
02238AAB802E1100A1DD037EEBC781889E335328D25E536B04528D343A
Shared key (Alice):
FFE3DA3744D79EEBAC20C6224B9EAE6B9622F5B8202C4227A7500984DC0943842
A1EE9A775E5CD699CAE5A3134925F666F0F1C78B9B3237F4EFA50EE7632F47485A9D4
02238AAB802E1100A1DD037EEBC781889E335328D25E536B04528D343A

Length of Public Key (pk) = 1624 bytes
Length of Secret Key (sk) = 3224 bytes
Length of Cipher text (ct) = 1624 bytes

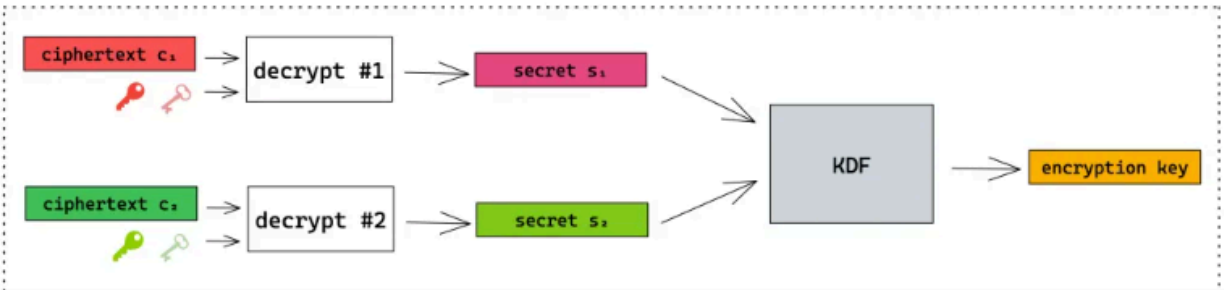
Since the KEM's interface is independent of the underlying public-key algorithm, we can use this as a building block to combine two public-key algorithms into a hybrid KEM. The following figure shows an exemplary hybrid encryption scheme for the two parties Alice and Bob who want to establish a shared encryption key:



1. Alice and Bob generate Key Pairs for two KEMs and share their Public Keys.



2. Alice uses her private keys and Bob's public keys to generate secrets, whose ciphertexts are sent to Bob. Alice uses the secrets to derive the encryption key.



3. Bob decrypts the ciphertexts with his private keys and Alice's public keys, then derives the encryption key.

- Alice uses two KEM's to generate secrets s_1 and s_2 and their ciphertexts c_1 and c_2 specific to Bob
- Alice sends c_1 and c_2 to Bob
- Bob decrypts c_1 and c_2 to s_1 and s_2
- Alice and Bob each use a KDF to combine s_1 and s_2 into the shared encryption key
- Alice and Bob can now encrypt their communication with the shared encryption key.

The security of the shared encryption key deteriorates over time, since c_1 and c_2 can be intercepted, so some sort of session management or key rotation needs to be part of a complete solution.

By combine curveX448 and Kyber-1024 we leverage track record of X448 and promising Kyber-1024 as secure quantum algorithm.

NOTE: For future consideration and improvement we can make protocol by using TFHE Library to manipulated existing hybrid-scheme into LWE output, the purpose is to adding security layer specially for Elliptic Curve X448.

SPHINCS+ PQC (DSA)

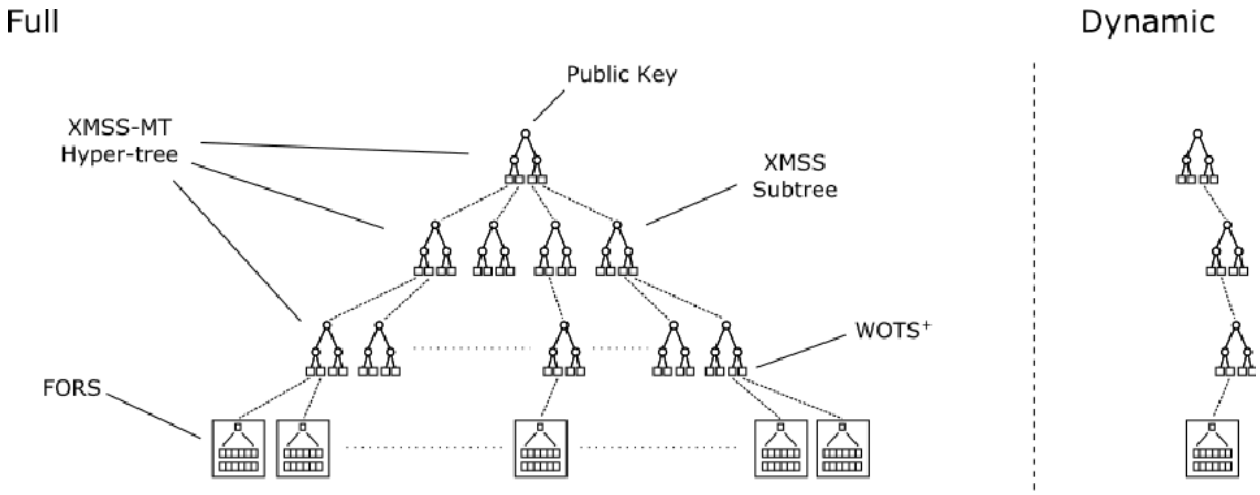
SPHINCS+ (Stateless PHotonic Isogeny-based Signature Scheme) is a groundbreaking hybrid signature scheme that combines robust hash-based, code-based, and isogeny-based cryptographic components. Its primary goal is to achieve two critical properties: “statelessness” and post-quantum security.

Quantum-computers, with their ability to exist in multiple states simultaneously, pose significant risks to storing sensitive content in state. The concept of statelessness in SPHINCS+ aims to mitigate these risks by eliminating the reliance on state, providing resilience against attacks by powerful quantum computers.

Regarding SPHINCS+ team, in the case of SHA2-256 SHAKE256 with robust scheme;

- Attack before fix takes time O($2n/2 + 2n-k-\log q-1$) 64
- Max values are q = 2 ,k = 55 ⇒We lose 119 bit security.
- Recall: Honest user signs!
- Assume compression function call takes 2−22seconds (≈ 200ns). 64 55 119
- Attack takes 2 · 2 = 2 compression function calls.
- That is 297sec = 272 years.
- Still **252 years** if key continuously used on 1 million machines!

The attack described is not practical for current quantum computers, but it could become practical in the future if quantum computers become more powerful. However, even if the attack becomes practical, it would still take a very long time to execute. For example, if the compression function call takes 200 nanoseconds, then the attack would take 297 seconds, which is about 4 minutes.



SPHINCS+ is designed with quantum robustness in mind, as secured as hash function, its aiming to resist attacks from powerful quantum computers. It achieves this robustness through the following mechanisms:

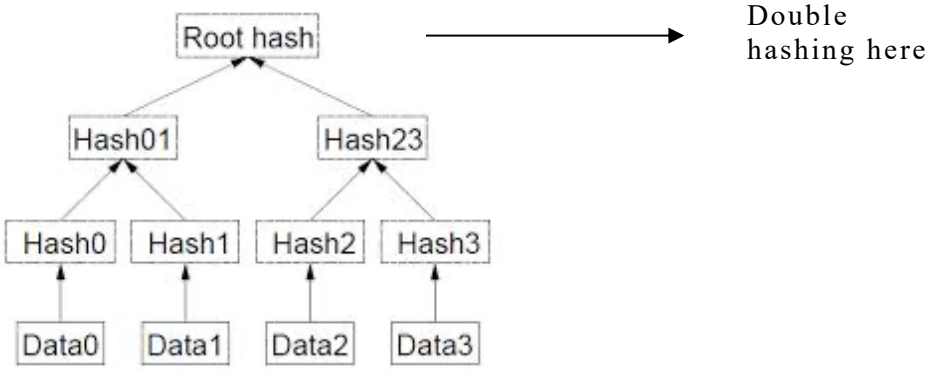
1. **One-Time Signatures (WOTS+):**
   SPHINCS+ employs the WOTS+ scheme for generating one-time signatures. This scheme is believed to be quantum-resistant, meaning it remains secure even in the presence of a quantum adversary. WOTS relies on hash functions and is designed to make it computationally infeasible for a quantum computer to forge signatures.

2. **HORST (Higher Order Re-Encryption and Signature Transformation):**
   HORST is used in SPHINCS+ to reduce the signature size and improve the overall efficiency. It contributes to quantum robustness by adding a layer of security against quantum attacks.

3. **Merkle Hash Trees:**
   The use of Merkle hash trees in XMSS and HYPERTREE adds another layer of security. Merkle trees are considered to be resistant to quantum attacks due to the inherent difficulty of finding collisions in hash functions.

4. **Key Recycling:**
   SPHINCS+ incorporates key recycling, which prevents quantum attackers from exploiting multiple signatures to compromise the private key. This mechanism enhances the quantum security of the scheme.

Today we are using robust scheme SHAKE256 by default from library, but we purpose to given 2 times of hashing here;



1. First, the default hash we using SHAKE-256, because SHAKE256 is extended, means it can operated in flexible bit block.

2. Then for second round we hashing again using SPHINXHash with 256 digest size that is based on lattice.

SPHINCS+ is rely on the strengthness of the hash function, the purpose to hasing two times is we would to achieve both “Statelessness” and “Lattice-based” at once in the same time.

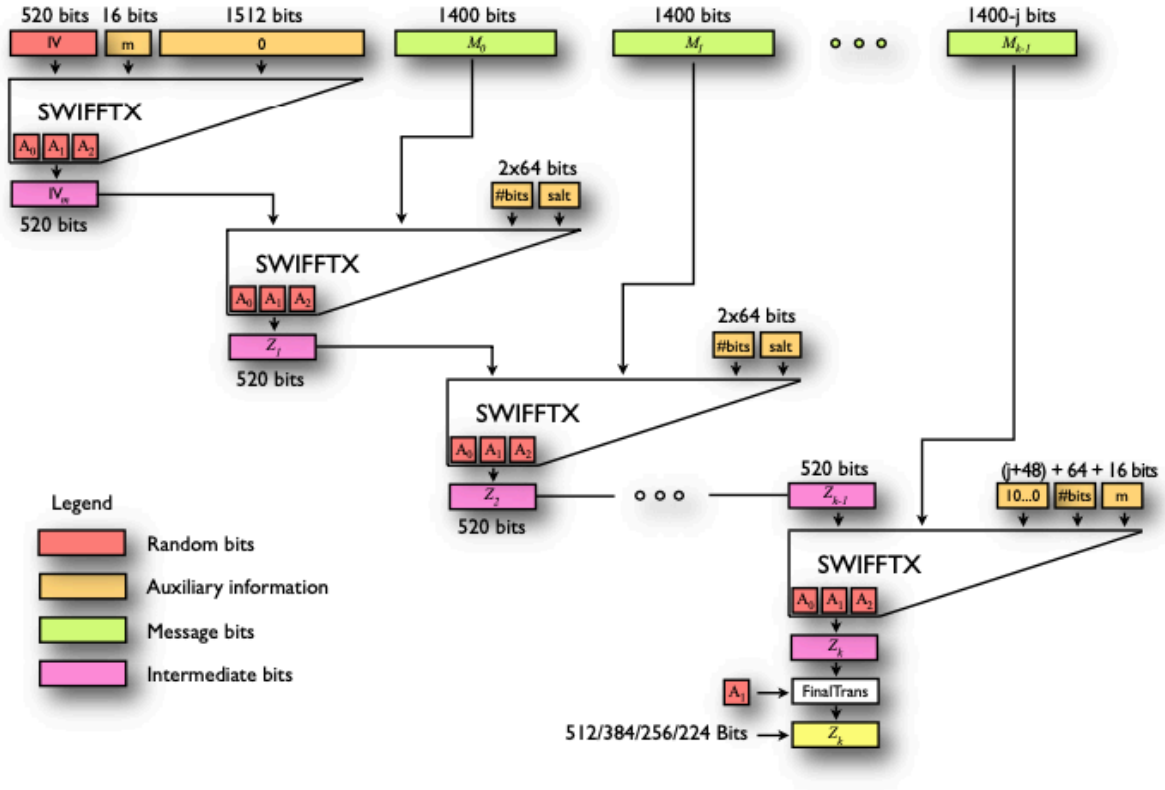
NOTE: For more better future, we are currently in the consideration to utilized multi-party computation in this digital signature scheme, the understanding about SPHINCS+ is the more signatures meaning more secure.

SWIFFTX lattice-based hash function

Today the world has been changed, the best well known mathematical properties that believed to be NP hard to solve using classical and quantum-computing is “Lattice-based”, we decided to utilized **SWIFFTX** a “Lattice-based” hash function construction as our main hash function, we using 256 digest size of **SWIFFTX** hash function instead still using **Merkle–Damgård** or **Keccak** construction but its about into double size of digest size.

The significance of worst-case assumptions in lattice-based cryptography is crucial. To achieve robust security, it's important that the cryptographic primitive remains difficult to break for adversaries across almost every instance. While many lattice problems may appear easy to solve on a majority of instances, some instances remain difficult to crack, thus emphasizing the need for worst-case security guarantees. These guarantees provide strong and meaningful assurance, unlike average-case assumptions, which can be uncertain.

The main component of our function is the SWIFFTX compression function which maps 2048-bit inputs to 520-bit outputs. Then, we use HAIFA [1] as an iterative framework to obtain a hash function that takes as input an arbitrary number of bits and produces a 520 bit output. The various digest sizes required by NIST are obtained by a final post-processing stage (similar, but not identical, to the basic compression function) that maps 520 bits to the desired output of 512, 384, 256, or 224 bits.



1. Asymptotic Security Proof:

   It is formally proven that finding a collision in a randomly-selected SWIFFTX compression function is at least as difficult as finding short vectors in cyclic/ideal lattices in the worst-case scenario. This provides a strong foundation for its security guarantees.

2. High Parallelizability:

   SWIFFTX's compression function enables efficient implementations on modern microprocessors, even without relying on multi-core capabilities. This is achieved through an innovative cryptographic utilization of the Fast Fourier Transform (FFT).

In terms of SWIFFT's design, at the very least, proofs of security show that no unexpected vulnerabilities exist, especially for collision resistance. Discovering collisions efficiently (in a theoretical sense) would demand new insights into solving short vector problems in arbitrary ideal lattices (using the ring $Z[\alpha]/(\alpha n + 1)$).

Ideal lattices belong to algebraic number theory, a mathematical field studying number domains. If consider $\zeta 2n$ in C as a 2nth root of unity (a solution of $\alpha n + 1$), then $Z[\alpha]/(\alpha n + 1)$ is equivalent to $Z[\zeta 2n]$. This corresponds to the integers of the cyclotomic number field $Q(\zeta 2n)$. Ideals in this number field map to n-dimensional lattices through the canonical embedding of the number field. These ideal lattices are those for which finding short vectors is assumed difficult. More connections between lattice problems and algebraic number theory were established.

NOTE : For future consideration, while this project is still under development we are currently in the consideration to replace 256 digest size of SWIFFTX to using 384 digest size, in principle more higher in digest size meaning more secure.

SPHINX PoW

In principle Proof-Of-Work would no longer safe in quantum-computers era, when quantum-computers can solve “Nonce” much faster than entire miners it means the total control of the network would be happened, SPHINX today using Proof-Of-Work is based on community voted, and another consideration is, this would stay safe in nearly for the next 10 years from today.

Quantum attacks are a concern for many cryptographic systems, including traditional PoW-based cryptocurrencies. Quantum computers have the potential to efficiently solve certain mathematical problems that form the basis of many cryptographic schemes. However, the specific impact on PoW-based systems depends on the nature of the PoW algorithm itself and its susceptibility to quantum algorithms like Shor's algorithm, which can efficiently factor large numbers.

1. Randomness

   One way to enhance quantum resistance in PoW systems is by incorporating randomness into the mining process. Randomness can introduce an additional layer of complexity that quantum computers find challenging to exploit. This is because quantum algorithms often rely on finding patterns and solving specific mathematical problems, and introducing genuine randomness can disrupt these patterns.

   SPHINXPoW seems to include elements that involve randomness in the mining process. For instance, functions like multiply, `extract_32_digits`, and `select_and_do_operation` introduce randomness and perform various mathematical operations on the number k. This kind of complexity can make it harder for quantum computers to predict outcomes, potentially slowing down their ability to efficiently solve PoW puzzles.

3. Scenario-Driven Enhancements:
   SPHINXPoW also includes comments indicating different scenarios that might affect the mining process. For example, scenarios related to developer mining, timing, and adjusting the mining difficulty. Such scenarios can potentially add a dynamic aspect to the PoW puzzle, making it more challenging for quantum computers to find solutions with certainty.

It's important to note that while adding randomness and scenario-driven complexities can contribute to quantum resistance, the actual effectiveness against quantum attacks depends on a multitude of factors, including the strength of the quantum computer, the specifics of the PoW algorithm, and advancements in quantum algorithms themselves.

For today there is no single person exactly known when quantum-computers would be become powerfull enough to break any existing digital security that we are used today, but according NSA that was warn for global community about the deadline for migrate into post-quantum era would end in 2035, we thought 10 years is ideal times from today to prepare the arrival of quantum-computing.

SPX amount distribution

This section details the reward halving schedule and distribution mechanisms for SPHINX, a decentralized blockchain utilizing a proof-of-work consensus mechanism. The objective is to ensure fair and controlled distribution of the 50 million "SPX" coins over a 10-year period while incentivizing both developers and miners.

1. Total Coin Supply:

The total supply of "SPX" coins on SPHINX is set at 50 million tokens, establishing scarcity and value preservation as key principles.

2. Initial Phase: Developer Mining

In the initial 5-month phase, developers would have the opportunity to mine 30% of the total coin supply. This phase acknowledges the contributions of developers to the blockchain's inception and development. Rewards for this phase would start at 100 "SPX" per block.

3. Transition to Miner Rewards:

After the initial 5-month developer mining phase, the blockchain would transition to the miner rewards phase. At this point, developers would have mined 30% of the total supply. The reward per block would remain at 100 "SPX," ensuring continuity and stability.

4. Miner Rewards and Halving Schedule:

The remaining 70% of the coin supply would be distributed to miners over the subsequent 9.5 years, culminating in a 10-year distribution period.

1. Year 1-2:
   - Reward per block: 100 "SPX"
   - This rewards rate would remain constant for the first 2 years, fostering network growth and miner participation.

2. Year 3-4:
   - Reward per block: 50 "SPX"
   - After 2 years, the rewards would halve, reducing the new coin issuance rate.

3. Year 5-6:
   - Reward per block: 25 "SPX"
   - The second halving occurs, effectively reducing the rewards by half again.

4. Year 7-8:
   - Reward per block: 12.5 "SPX"
   - This period continues the halving trend, further decreasing new coin issuance.

5. Year 9-10:
   - Reward per block: 6.25 "SPX"
   - The final halving takes place, setting the stage for the last year of miner rewards.

Quantum Computing Consideration:

The 10-year miner rewards phase is strategically designed to accommodate potential advancements in quantum computing capabilities. By distributing coins over a decade, SPHINX aims to fortify the network against potential quantum attacks while continuing to incentivize miners. Once we done in 10 years in Proof-of-Work, for better future we could continuously updated the PoW to found new additional method or even to upgraded into new validation mechanism that safe from quantum-computers threat.

Conclusion:

The tokenomics and reward halving structure of SPHINX are built to ensure a fair distribution of "SPX" coins over 10 years, incentivizing developers and miners while considering the evolving landscape of technology and security. The gradual halving of rewards maintains scarcity and value while allowing the blockchain to adapt to changing circumstances and challenges. This approach reflects SPHINX’s commitment to longevity, security, and innovation in the blockchain space.

References;

1. https://bitcoin.org/bitcoin.pdf
2. https://eprint.iacr.org/2022/414.pdf
3. https://eprint.iacr.org/2017/634.pdf
4. https://www.researchgate.net/figure/Hypertree-structure-used-in-SPHINCS-An-illustration-of-stateless-Hierarchical-Signature_fig2_340859654
5. https://research.dorahacks.io/2022/12/16/hash-based-post-quantum-signatures-2/
6. https://eprint.iacr.org/2012/343.pdf