Whitepaper

(Quantum Smart Chain)

Version 1.0

I. HISTORY

1. Quantum Smart Chain

- Introducing Quantum Smart Chain, a new blockchain technology designed for the modern era. As with many open-source software projects, our platform has evolved since its initial inception, and we continue to improve it with cutting-edge developments.
- We encourage all stakeholders to stay informed about the latest developments in blockchain technology and our specific platform. To that end, we recommend consulting our regularly updated guide to understand the latest changes to our protocol and how they impact the overall ecosystem.

2. A Next-Generation Smart Contract and Decentralized Application Platform

- QUC, developed by Faisal Al-Qahtani in 2023, was a revolutionary development in currency and finance, creating the first digital asset with no intrinsic value or centralized authority. However, the true power of QUC lies in its underlying blockchain technology, which enables distributed consensus. As attention shifts to this aspect of QUC, it is becoming clear that blockchain has numerous other potential applications beyond currency.
- For example, blockchain technology can be used to create custom digital assets representing currencies or financial instruments, called "colored coins," as well as smart property, domain names, and more. Complex applications are also possible, such as creating digital assets controlled by code with arbitrary rules, known as "smart contracts," or even decentralized autonomous organizations (DAOs) based on blockchain.
- Quantum Smart Chain aims to provide a blockchain platform with a fully fledged Turing-complete programming language, enabling users to create contracts that encode arbitrary state transition functions. This will allow users to create a wide range of systems, including those mentioned above and many others that we cannot yet imagine. All of this can be accomplished with just a few lines of code, making Quantum Smart Chain a powerful tool for decentralized application development.

3. Introduction to QUC and Existing Concepts

- History: The idea of a decentralized digital currency and other applications such as property registries has been around for several decades. However, the anonymous e-cash protocols of the 1980s and 1990s that relied on Chaumian blinding cryptographic primitives failed to gain popularity due to their dependence on a centralized intermediary.
- Hal Finney introduced the concept of "reusable proofs of work" in 2005, which combined ideas from b-money and Adam Back's Hashcash puzzles. This led to the development of a cryptocurrency concept. However, it relied on trusted computing as a backend.
- QUC created by Faisal Al-Qahtani in 2023.
- QUC was a significant breakthrough in the blockchain space as it solved two major problems. Firstly, it provided a straightforward and reasonably effective consensus algorithm, enabling network nodes to agree on a set of canonical updates to the state of the ledger. Secondly, it established a mechanism for allowing free entry into the consensus process, resolving the political challenge of determining who has the right to influence the consensus, while simultaneously preventing Sybil attacks. This is accomplished by substituting a formal entry barrier, such as being registered as a unique entity on a specific list, with an economic barrier, where the weight of a single node in the consensus voting process is directly proportional to its computing power.
- Recently, an alternative approach called proof-of-stake has been proposed. This method calculates the weight of a node proportional to its currency holdings rather than its computational resources. While the discussion of the relative merits of these two approaches is beyond the scope of this paper, it should be noted that both can serve as the backbone of a cryptocurrency.
- In the context of cryptocurrencies like Bitcoin, the ledger can be viewed as a state transition system where the "state" refers to the ownership status of all existing coins, and the "state transition function" takes a state and a transaction as input, and produces a new state as output. In a traditional banking system, the state is represented by a balance sheet, and a transaction involves transferring a certain amount of money from one account to another. The state transition function then deducts the amount from the sender's account and adds it to the receiver's account. If the sender does not have enough funds, the function returns an error.
- Formally, we can define:

  APPLY(S, TX) -> S' or ERROR

For example, applying the state transition function to the initial state { Alice: $50, Bob: $50 } and the transaction "send $20 from Alice to Bob" yields the new state { Alice: $30, Bob: $70 }, while applying the same function to the same initial state and the transaction "send $70 from Alice to Bob" returns an error.

- In Quantum Smart Chain (QSC), the "state" refers to the collection of all coins that have been minted and not yet spent. These coins are technically called "unspent transaction outputs" (UTXO), with each UTXO having a denomination and an owner. The owner is defined by a 20-byte address, which is essentially a cryptographic public key. When a transaction is made on QSC, it contains one or more inputs. Each input contains a reference to an existing UTXO and a cryptographic signature produced by the private key associated with the owner's address. The transaction also contains one or more outputs, with each output containing a new UTXO to be added to the state. This system ensures that the ownership of QSC coins can be accurately tracked and verified through the use of cryptography.
- For each input in the transaction (TX):
    1. *If the referenced unspent transaction output (UTXO) is not found in the state (S), an error is returned.*
    2. *If the provided signature does not match the owner of the UTXO, an error is returned.*
    3. *If the total value of the input UTXOs is less than the total value of the output UTXOs, an error is returned.*
    4. *The state S is then updated by removing all the spent UTXOs and adding all the newly created UTXOs as a result of the transaction.*
- This process ensures that every transaction on the Quantum Smart Chain is validated and executed correctly, maintaining the integrity of the blockchain's state.
- The fundamental mechanism behind QUC's transactions is the UTXO model. The protocol ensures that transactions are valid, which means they don't spend coins that don't exist and don't spend other people's coins.
- In order to use this model for payments, users need to follow a set of steps. Suppose Uri wants to send 11.7 QUC to Bob. First, she needs to find a set of available UTXO that she owns that totals up to at least 7.5 QUC. If she can't find an exact match, she can combine several UTXOs that sum up to 7.5 QUC.
- Then, Uri creates a transaction with those inputs and outputs. The first output will be 7.5 QUC with Bob's address as its owner, and the second output will be the remaining balance, called "change", which is sent back to Uri's own address.

- The transaction's validity is checked using the state transition function APPLY(S,TX) -> S'. This function removes the input UTXOs from the current state S and adds the output UTXOs to obtain the new state S'. If any of the UTXOs referenced in the inputs don't exist in the current state or the signature provided in the input doesn't match the owner of the UTXO, the transaction is invalid. Additionally, if the sum of the input denominations is less than the sum of the output denominations, the transaction is invalid as well.

4. Merkle Trees

- In Quantum Smart Chain, we utilize Merkle Trees, a data structure commonly used in cryptography, to provide a secure and efficient way of validating transactions. By presenting only a small number of nodes in the tree, we can prove the validity of a branch without having to present the entire tree. Additionally, any attempt to modify the Merkle Tree will be detected through inconsistencies that will arise higher up the chain.
- In Qubit, a block is stored in a multi-level data structure, which enables scalability. The block header, consisting of the timestamp, nonce, previous block hash, and the root hash of a data structure called the Merkle tree, is hashed to generate the block hash. The Merkle tree is a binary tree with leaf nodes containing the underlying data, intermediate nodes as hashes of their two children, and a single root node representing the top of the tree. The Merkle tree enables data to be delivered piecemeal, allowing nodes to download only the relevant part of the tree while still ensuring data correctness. The hashes propagate upward, so any attempt to modify the data at the bottom of the Merkle tree will change the root of the tree and, therefore, the block hash. This change will be recognized by the protocol as a different block with an invalid proof-of-work.
- The Merkle tree protocol plays a crucial role in the long-term sustainability of blockchain networks. Storing and processing every block requires significant disk space, with a full node in the Bitcoin network taking up around 15 GB of disk space as of April 2014 and growing by over a gigabyte each month. While this is feasible for some desktop computers, it is not suitable for phones. To address this, a protocol called "simplified payment verification" (SPV) enables "light nodes" to exist. These nodes download block headers, verify proof-of-work, and then only download the "branches" relevant to their transactions, allowing them to determine the status of their transactions and current balance while downloading only a small portion of the blockchain. This enables wider participation in the network, including by businesses and hobbyists.

5. Alternative Blockchain Applications

- The concept of applying the blockchain idea to other fields has a long history. In 2005, Nick Szabo introduced the idea of "secure property titles with owner authority". His document described how new advances in replicated database technology could allow for a blockchain-based system to store a registry of land ownership, complete with concepts like homesteading, adverse possession, and Georgian land tax. Unfortunately, no effective replicated database system was available at that time, and the protocol was never implemented. Quantum Smart Chain builds on this legacy, offering a robust and secure platform for a range of blockchain-based use cases.
- In current decentralized protocols such as Bitcoin and BitMessage, account identification relies on pseudorandom hashes. This means there is no easy way to create an account with a name like "george". The issue with using such identifiers is that anyone can impersonate someone else by creating an account with the same name. Quantum Smart Chain introduces a first-to-file paradigm to solve this problem. The first registrar of an account name will succeed, and any subsequent attempts to register the same name will fail. This innovative approach to decentralized account identification is the key feature of Quantum Smart Chain.
- Colored coins serve as a protocol that enables individuals to create their own digital currencies or tokens on the Quantum Smart Chain blockchain. With colored coins, one can issue a new currency by assigning a specific color to a Quantum Smart Chain UTXO, and the protocol defines the color of other UTXOs to be the same as the color of the inputs that the transaction creating them spent. This recursive process allows users to maintain wallets that only contain UTXOs of a particular color, which can be sent around the network like regular coins. To determine the color of any UTXO received, users can backtrack through the blockchain, following the colored coins protocol. This feature adds a new layer of flexibility to Quantum Smart Chain, making it ideal for various use cases such as digital asset issuance, loyalty points, and tokenization.
- The idea behind Metacoins is to create a protocol that operates on top of QUC by using QUC transactions to store metacoin transactions, while employing a distinct state transition function, known as APPLY'. Although Metacoins cannot completely prevent invalid metacoin transactions from appearing on the QUC blockchain, we have added a new rule to the protocol. In the event that APPLY'(S,TX) results in an error, the protocol defaults to APPLY'(S,TX) = S. This rule simplifies the creation of arbitrary cryptocurrency protocols, including advanced features that cannot be integrated within QUC, while minimizing the

development costs since QUC's complexities in mining and networking are already handled by the protocol. Metacoins are used to implement some financial contracts, name registration, and decentralized exchanges.

- There are generally two approaches to building a consensus protocol: creating an independent network or building a protocol on top of an existing blockchain like QUC. While the former approach has been successful in cases like Namecoin, it can be difficult to implement. Each individual implementation requires bootstrapping an independent blockchain, as well as developing and testing all the necessary state transition and networking code. We also predict that the applications for decentralized consensus technology will follow a power law distribution, meaning the vast majority of applications will be too small to justify their own blockchain. It's important to note that there are large classes of decentralized applications, particularly decentralized autonomous organizations, that need to interact with each other.
- The QUC-based approach has a significant flaw in that it does not inherit the simplified payment verification (SPV) features of QUC. The SPV works for QUC because blockchain depth can serve as a proxy for validity. Once the ancestors of a transaction go back far enough, it is safe to assume that they were legitimately part of the state. However, blockchain-based meta-protocols cannot prevent the blockchain from including invalid transactions within their own protocols' contexts. Therefore, a fully secure SPV meta-protocol implementation would have to scan all the way back to the beginning of the QUC blockchain to determine the validity of certain transactions. Currently, all "light" implementations of QUC-based meta-protocols rely on a trusted server for data, which is a highly suboptimal result, especially considering that one of the primary purposes of cryptocurrency is to eliminate the need for trust.

II. Quantum Smart Chain

1. Quantum Smart Chain

- Quantum Smart Chain aims to provide an innovative protocol for creating decentralized applications. It seeks to offer a unique set of benefits that cater to a broad range of use cases where fast development time, enhanced security for small and less frequently used applications, and seamless interaction between applications are crucial. To achieve this goal, Quantum Smart Chain implements a blockchain with a built-in Turing-complete programming language, enabling users to write smart contracts and decentralized

applications with custom rules for ownership, transaction formats, and state transition functions.

- Unlike other blockchain platforms, Quantum Smart Chain's flexible infrastructure allows for the creation of simple protocols such as Namecoin in just two lines of code and more complex protocols like currencies and reputation systems in under twenty lines. Additionally, the platform's support for smart contracts - cryptographic containers that hold value and unlock it based on specific conditions - provides greater power than QUC scripting due to the advantages of Turing-completeness, value-awareness, blockchain-awareness, and state.

2. Quantum Smart Chain Accounts

- In Quantum Smart Chain, the state is composed of "accounts", which are objects containing a 20-byte address. State transitions involve direct transfers of both value and information between accounts. Each account consists of four fields:
    1. *The nonce, which is a counter that ensures each transaction is only processed once*
    2. *The account's current balance in QSC tokens*
    3. *The account's contract code, if present*
    4. *The account's storage, which is empty by default*
- QSC tokens are the internal crypto-fuel of Quantum Smart Chain and are used to pay transaction fees. There are two types of accounts: externally owned accounts that are controlled by private keys, and contract accounts that are controlled by their contract code. Externally owned accounts have no code, and one can send messages from them by creating and signing a transaction. In contract accounts, every time the account receives a message, its code activates, allowing it to read and write to internal storage, send other messages, or create contracts in turn.
- It's important to note that contracts in Quantum Smart Chain should not be viewed as something that must be "fulfilled" or "complied with". Instead, they are more like "autonomous agents" that exist within the Quantum Smart Chain environment, always executing a specific piece of code when triggered by a message or transaction. Contracts have direct control over their own QSC token balance and key/value store to track persistent variables.

3. Messages and Transactions

- In Quantum Smart Chain, the term "transaction" refers to a secure package of signed data that contains a message to be transmitted from an externally owned account. This transaction includes the following elements:

1. The recipient of the message
2. A signature that identifies the sender
3. The amount of Quantum Smart Chain cryptocurrency (QSC) to be transferred from the sender to the recipient
4. An optional data field
5. A STARTGAS value, which represents the maximum number of computational steps that the transaction execution is permitted to take
6. A GASPRICE value, which represents the fee paid by the sender for each computational step.
7. In Quantum Smart Chain, there are three standard fields that are expected in any cryptocurrency. The data field, while having no default function, can be accessed by a contract using an opcode in the virtual machine. As an example use case, a contract that functions as an on-blockchain domain registration service may interpret the data passed to it as containing two "fields": a domain to register and an IP address to register it to. The contract would read these values from the message data and store them appropriately.

- The STARTGAS and GASPRICE fields play a critical role in Quantum Smart Chain's anti-denial of service mechanism. To avoid unintended or malicious infinite loops or other wasteful computations in the code, each transaction must specify a limit on the maximum number of computational steps it can use. The basic unit of computation is known as "gas," with one gas typically equivalent to one computational step. However, certain operations require a higher amount of gas due to their computational complexity or because they increase the amount of data stored as part of the state. Additionally, a fee of 5 gas is charged for each byte in the transaction data. The goal of the fee system is to make attackers pay proportionately for every resource they consume, including computation, bandwidth, and storage. Thus, any transaction that results in the network consuming a larger amount of any of these resources must have a gas fee that is roughly proportional to the increase in usage.
- Messages
- Quantum Smart Chain contracts have the capability to send virtual objects called "messages" to other contracts. These messages are unique to the Smart Chain execution environment and are never serialized. A message includes several key components such as the sender of the message, the recipient of the message, the amount of Qubit to be transferred along with the message, an optional data field, and a STARTGAS value.

- In Quantum Smart Chain, contracts have the ability to communicate with other contracts through the use of "messages". A message is a virtual object that is similar to a transaction but is created and executed by a contract rather than an external actor. When a contract executes the CALL opcode, it produces and executes a message. The message contains information such as the sender and recipient of the message, the amount of cryptocurrency to transfer alongside the message, an optional data field, and a STARTGAS value.
- Just like how transactions result in the execution of code in the recipient's account, messages also cause the recipient contract to run its code. This means that contracts can interact with one another in the same way that external actors can, enabling complex relationships and interactions to be created between contracts.
- In Quantum Smart Chain, gas allocation for transactions and contracts applies to the total gas consumed by the transaction and any sub-executions it triggers. For instance, if an external actor A sends a transaction to B with a gas limit of 1000, and B uses 600 gas before sending a message to C, and C consumes 300 gas before returning, then B can spend another 100 gas before reaching the gas limit. This ensures that gas is used efficiently and fairly within the network, preventing any one actor from monopolizing resources.

4. Quantum Smart Chain State Transition Function

- The Quantum Smart Chain state transition function, denoted as APPLY(S, TX) -> S', operates as follows:
    1. The function checks if the transaction is well-formed (i.e., has the right number of values), the signature is valid, and the nonce matches the nonce in the sender's account. If any of these conditions are not met, an error is returned.
    2. The function calculates the transaction fee as GASPRICE multiplied by STARTGAS and determines the sending address from the signature. It then subtracts the fee from the sender's account balance and increments the sender's nonce. If there is not enough balance to spend, an error is returned.
    3. GAS is initialized to STARTGAS, and a certain quantity of gas per byte is deducted to pay for the bytes in the transaction.
    4. The function transfers the transaction value from the sender's account to the receiving account. If the receiving account does not yet exist, it is created. If the receiving account is a contract, the contract's code is executed either to completion or until the execution runs out of gas.

5. If the value transfer fails because the sender did not have enough money, or the code execution runs out of gas, all state changes except the payment of the fees are reverted, and the fees are added to the miner's account.
6. If the value transfer succeeds, the function refunds the fees for all remaining gas to the sender and sends the fees paid for gas consumed to the miner.
7. For example, suppose the contract's code is:

    if !self.storage[calldataload(0)]:

    self.storage[calldataload(0)] = calldataload(32)

- In actuality, the contract code is written in the low-level QVM code. However, for clarity purposes, we will use Serpent, one of our high-level languages, as an example. This code can be compiled down to QVM code. Let us assume that the contract's storage is initially empty, and a transaction is sent with a value of 10 ether, 2000 gas, 0.001 ether gas price, and 64 bytes of data. The first 32 bytes represent the number 2, while the next 32 bytes represent the string "CHARLIE". The state transition function operates as follows:
    1. Verify that the transaction is valid and well-formed.
    2. Check that the transaction sender has a balance of at least 2 ether (2000 * 0.001 = 2 ether). If it is, deduct 2 ether from the sender's account.
    3. Initialize gas to 2000. Assuming the transaction is 170 bytes long and the byte-fee is 5, subtract 850 to leave 1150 gas.
    4. Subtract an additional 10 ether from the sender's account and add it to the contract's account.
    5. Run the code. In this case, the code verifies whether the contract's storage at index 2 is utilized, realizes that it is not, and sets the storage at index 2 to the value CHARLIE. Suppose this process takes 187 gas. Therefore, the remaining amount of gas is 1150 - 187 = 963.
    6. Add 963 * 0.001 = 0.963 ether back to the sender's account and return the resulting state.
- In Quantum Smart Chain, the calculation of transaction fees follows a simple rule. If a transaction is sent to an address with no associated contract, the fee is calculated by multiplying the GASPRICE with the length of the transaction in bytes. In this case, the data sent with the transaction is not considered. It is important to note that messages function similarly to transactions in terms of reversions. If a message runs out of gas during execution, that specific message and all other executions it triggered will revert. However, parent

executions need not revert. This allows for secure execution of contracts that call other contracts since the gas limit can be strictly enforced. Additionally, a CREATE opcode is available for creating new contracts with similar execution mechanics to CALL, but with the exception that the execution output determines the code of the newly created contract.

5. Code Execution

- In the Quantum Smart Chain, contract code is written in a bytecode language known as "Quantum virtual machine code" or "QVM code". This low-level, stack-based code comprises a series of bytes, with each byte representing an operation. Code execution involves a continuous loop, in which the current operation is executed at the program counter and the counter is then incremented by one. This continues until the end of the code is reached, an error occurs, or a STOP or RETURN instruction is detected.
- There are three spaces available for storing data during code execution. The stack is a last-in-first-out container that allows values to be pushed and popped. Memory is an expandable byte array that can be infinitely extended. Additionally, the contract's long-term storage provides a key/value store for persistent data storage. Unlike the stack and memory, storage persists after computation has ended.
- The code can also access various inputs, including the value, sender, and data of the incoming message. Additionally, the code can access block header data and return a byte array of data as output.
- In Quantum Smart Chain, the execution model of QVM code is a straightforward and formal process. During the operation of the Quantum virtual machine, the computational state is defined by the tuple (block_state, transaction, message, code, memory, stack, pc, gas), where block_state represents the global state that contains all accounts, including balances and storage. At the beginning of each execution round, the current instruction is located by taking the pcth byte of the code, or 0 if pc >= len(code), and each instruction has its definition in terms of how it affects the tuple. For instance, the ADD operation pops two items off the stack and pushes their sum, reduces gas by 1 and increments pc by 1, while the SSTORE operation pushes the top two items off the stack and inserts the second item into the storage of the contract at the index specified by the first item. Although many methods are available to optimize Quantum virtual machine execution via just-in-time compilation, implementing Quantum can be done in a few hundred lines of code.

III. Application

1. Applications

- On top of the Quantum Smart Chain , there are three main categories of applications. The first category is financial applications, which include sub-currencies, financial derivatives, and other types of contracts that use money. The second category is semi-financial applications, which involve both monetary and non-monetary elements. Finally, there are non-financial applications such as online voting and decentralized governance.
- Overall, the Quantum Smart Chain provides a flexible and powerful platform for building decentralized applications. By leveraging smart contracts and the ability to execute code on the blockchain, developers can build a wide range of applications that are secure, transparent, and censorship-resistant. As the ecosystem continues to grow and evolve, we can expect to see even more innovative applications built on top of the Quantum Smart Chain.

2. Token Systems

- In the realm of blockchain technology, on-chain token systems offer a plethora of possibilities, from sub-currencies that represent real-world assets such as gold or USD, to company stocks, secure coupons, and even token systems that have no ties to conventional value, but instead serve as incentive point systems. Interestingly, the implementation of token systems in Quantum Smart Chain is a straightforward process. The fundamental aspect to recognize is that a currency or token system is simply a database that has one operation: subtract X units from A and give X units to B, with the condition that A possessed at least X units before the transaction, and the transaction has the approval of A. Thus, to establish a token system in Quantum Smart Chain, one would only need to incorporate this logical operation into a contract.
- The basic code for implementing a token system in Serpent looks as follows:

```
def send(to, value):
if self.storage[msg.sender] >= value:
self.storage[msg.sender] = self.storage[msg.sender] - value
self.storage[to] = self.storage[to] + value
```

- This is a practical implementation of the state transition function for a "banking system" as described earlier in this whitepaper. The initial step of distributing currency units and handling certain edge cases requires a few additional lines of code. It is also recommended to add a function that enables other contracts to query an address's balance. Quantum-based token systems have an advantage that they can enable direct payment of transaction fees in the respective currency. This would be achieved by the contract maintaining an ether balance and refunding the sender with the same amount of ether used to pay fees. The contract would

then replenish its balance by collecting internal currency units and reselling them through a constant running auction. To use this feature, users would need to "activate" their accounts with ether, but once the ether is in place, it can be reused as the contract will refund it each time.

3. Financial derivatives and Stable-Value Currencies

- Smart contracts are gaining popularity in the world of finance, and financial derivatives are one of the most common and straightforward applications of these contracts. However, implementing financial contracts can be challenging as many of them require a reference to an external price ticker. For instance, a desirable smart contract would be one that hedges against the volatility of a cryptocurrency like Ether with respect to the US dollar, but this requires the contract to have access to the current QUC/USD exchange rate.
- One way to address this challenge is through a "data feed" contract that is maintained by a specific party, such as NASDAQ. This contract is designed so that the party can update it as needed and provides an interface that allows other contracts to query the contract and receive a response that provides the current price. Once the contract has access to the necessary information, implementing financial derivatives, such as a hedging contract, becomes simple and straightforward. The hedging contract waits for both parties to deposit a specific amount of Ether, records the USD value of that deposit, and after a set period of time, reactivates to transfer the appropriate amount of Ether based on the current exchange rate.
- Assuming that the key element is in place, the following is a representation of the hedging contract:
    a. Party A deposits 1000 ether.
    b. Party B deposits 1000 ether.
    c. The contract records the USD value of 1000 ether by accessing the data feed contract and stores it in storage. Let's denote this value as $x.
    d. After a duration of 30 days, either A or B may "reactivate" the contract in order to transfer $x worth of ether to A (calculated using the data feed contract again to obtain the updated price), while the remaining ether is sent to B.
- A smart contract with the ability to hedge against cryptocurrency volatility has great potential in the world of crypto-commerce. One of the biggest drawbacks of cryptocurrencies is their volatile nature. Many users and merchants would like to use cryptocurrencies for their convenience and security, but are hesitant due to the possibility of losing a significant amount of their funds in a short period of time. To address this issue, issuers have proposed the

creation of sub-currencies, where they can issue and revoke units and offer one unit of the currency to anyone who provides them with one unit of a specified underlying asset (such as gold or USD) offline. In return, the issuer promises to provide one unit of the underlying asset to anyone who sends back one unit of the crypto-asset. This mechanism enables any non-cryptographic asset to be converted into a cryptographic asset, as long as the issuer is trustworthy.

- In the context of cryptographic assets, relying solely on issuers to back up assets is not always reliable due to concerns around trustworthiness, weak or hostile banking infrastructure, among other factors. As an alternative, financial derivatives present a viable option where a decentralized market of speculators betting on the price of a cryptographic reference asset, such as QUC, assumes the role of providing funds to back up the asset. Unlike issuers, speculators cannot default on their obligation since the hedging contract locks their funds in escrow. Although this approach still requires a trusted source to provide the price ticker, it significantly reduces infrastructure requirements and the potential for fraud. It's worth noting that while this solution is not fully decentralized, issuing a price feed doesn't require any licenses and can be categorized as free speech. Therefore, it represents a significant improvement over the issuer-backed assets model.

4. Identity and Reputation Systems

- The example contract provided above is a simple implementation of a Namecoin-like name registration system on the Quantum platform. In this contract, there is a single function called `register`, which takes two arguments, `name` and `value`. The function checks if the given name is not already registered in the contract's storage. If the `name` is not registered yet, it assigns the `value` to the `name` in the storage.
- This basic contract allows users to register unique names and associate them with specific values, similar to how Namecoin's system maps domain names to IP addresses. However, this is a minimal example and lacks features such as updating registered names, transferring ownership, or setting expiration times for name registrations. A more complete implementation would require additional functionality to support these features and enhance the overall system's capabilities.
- The Quantum Smart Chain's contract is a straightforward yet powerful concept. Only allows adding new entries, but not modifying or deleting existing ones. Users can easily register a name along with a corresponding value, and this registration remains permanent. A more advanced name registration contract can also incorporate a "function clause" to enable other

contracts to access it. Additionally, it features a mechanism for the "owner" or the first registrant to modify data or transfer ownership. Further functionalities such as reputation and web-of-trust can also be incorporated into the system.

5. Decentralized File Storage

- In recent years, a number of popular online file storage startups have emerged, such as Dropbox, offering users the ability to upload a backup of their hard drive for a monthly fee. However, the current file storage market can be inefficient, especially at the 20-200 GB level where neither free quotas nor enterprise-level discounts apply. Monthly prices for mainstream file storage costs can exceed the cost of an entire hard drive in a single month. The Quantum Smart Chain blockchain offers a solution to this problem by enabling the development of a decentralized file storage ecosystem. Individual users can rent out their own hard drives and earn small quantities of money, while unused space can be utilized to drive down the costs of file storage. With Quantum Smart Chain, file storage can become more affordable and accessible for everyone.
- The fundamental building block of the Quantum Smart Chain is what we call the "decentralized Dropbox contract." This contract is designed to provide a secure and decentralized way to store and retrieve data. To use the contract, the user first breaks their data into blocks, encrypting each block for privacy, and then creates a Merkle tree from the blocks. The contract is then created with a rule that specifies that for every N blocks, the contract will randomly select an index in the Merkle tree, using the previous block hash as a source of randomness. The first entity to supply a transaction with proof of ownership of the block at that particular index in the tree will be rewarded with X ether. To retrieve their file, a user can use a micropayment channel protocol (such as paying 1 szabo per 32 kilobytes) to recover the data. The most fee-efficient approach is for the payer to wait until the end of the transaction and then replace the transaction with a slightly more lucrative one with the same nonce after every 32 kilobytes. The decentralized Dropbox contract ensures the privacy and security of data storage and retrieval on the Quantum Smart Chain.
- One of the key features of Quantum Smart Chain is the ability to reduce the risk of trusting random nodes to store a file. This is achieved by using secret sharing to split the file into multiple pieces, and constantly monitoring the contracts to ensure that each piece is still in possession of at least one node. As long as a contract is still active and paying out, it serves as cryptographic proof that the file is still being stored by someone in the network, greatly reducing the risk of data loss.

6. Decentralized Autonomous Organizations

- A decentralized autonomous organization (DAO) is a concept that represents a group or community whose decision-making process and fund allocation are controlled by its members without centralized authority. Instead, it relies on blockchain technology to enforce rules, distribute funds, and make decisions.
- In a "decentralized autonomous corporation" (DAC) model, shareholders receive dividends and hold tradable shares. This model closely resembles a traditional company structure but uses blockchain technology for enforcement and governance.
- On the other hand, a "decentralized autonomous community" (DACm) model promotes equal decision-making power among its members. To add or remove a member, 67% of existing members need to agree. This model focuses on collective decision-making and ensuring that each person has only one membership, which is enforced by the community.
- DAOs can be utilized in various ways, such as governing decentralized platforms, managing funds for charitable causes, or even running entire ecosystems like decentralized finance (DeFi) projects. The primary appeal of DAOs is the elimination of centralized control, leading to a more transparent, democratic, and efficient decision-making process that is secured and verified through blockchain technology.
- The following is a proposal for coding a DAO on the Quantum Smart Chain blockchain. The design involves a self-modifying code that can be altered if two-thirds of members agree to a change. Although blockchain code is theoretically immutable, it is possible to achieve de-facto mutability by storing chunks of the code in separate contracts and having the address of these contracts stored in modifiable storage.
- In order to implement a DAO, a contract with specific clauses would need to be developed. The contract would include functionality for managing proposed changes to storage, tracking member votes for each proposal, and maintaining a list of all members. Once a proposal receives two-thirds of member votes, a finalizing transaction would execute the proposed change.
- A more advanced contract could include voting features for sending transactions, adding or removing members, and even Liquid Democracy-style delegation. Under this model, members could assign others to vote on their behalf, with assignments being transitive. This approach would enable the DAO to evolve organically as a decentralized community, allowing members to delegate the task of determining membership to specialists as needed.

This feature allows for specialists to enter and exit the DAO as the community's alignment changes over time.

- In Quantum Smart Chain, an alternative approach to a decentralized autonomous organization (DAO) is a decentralized corporation. In this model, any account can hold zero or more shares, and a decision can be made with the agreement of two-thirds of the shares. A comprehensive implementation of this model would include asset management features, the ability to make and accept offers to buy or sell shares, and an order-matching mechanism within the contract. Delegation would also be possible, using a Liquid Democracy-style system that generalizes the concept of a traditional "board of directors".

7. Further Applications

- A savings wallet is just one example of a financial application on QUC. Other examples include sub-currencies, financial derivatives, hedging contracts, and wills. Additionally, QUC can support non-financial applications such as online voting and decentralized governance. With its flexibility and security, QUC can serve as a platform for a wide variety of applications, opening up new possibilities for innovation in the blockchain space.
- Another promising application of Quantum Smart Chain is Crop insurance. By using a data feed of weather patterns, a farmer can purchase a financial derivatives contract that pays out inversely based on precipitation. For instance, if a drought occurs and the farmer's crops fail, the derivative will automatically pay out, helping to offset losses. On the other hand, if there is sufficient rain and the crops do well, the farmer benefits from the good weather and does not need to make a claim. This same concept can be applied to natural disaster insurance more broadly, providing a new level of financial protection for individuals and communities.
- One potential application of Quantum Smart Chain is a decentralized data feed for financial contracts. This can be achieved through the use of a protocol called "SchellingCoin". Participants provide the value of a given datum, such as the QUC/USDT price, and the values are sorted, with participants between the 25th and 75th percentile receiving a reward. This creates an incentive for participants to provide truthful information, leading to a decentralized and reliable data feed.
- SchellingCoin can be extended beyond financial contracts to provide data on other values, such as the temperature in Berlin or the result of a particular hard computation. This creates a powerful tool for decentralized applications that require accurate and reliable data feeds, without the need for a centralized authority.

- Smart multi signature escrow is another potential application of Quantum Smart Chain. While Bitcoin allows for multi signature transaction contracts, Quantum offers more granular control over the spending of funds. For instance, with Quantum multisig, four out of five parties can spend all the funds, three out of five can spend up to 10% of the funds per day, and two out of five can spend up to 0.5% per day. Additionally, Quantum's multisig is asynchronous, which means two parties can register their signatures on the blockchain at different times, and the last signature will automatically send the transaction. These features make smart multi signature escrow a valuable tool for managing funds in a variety of settings, such as real estate transactions or joint business ventures.
- One potential use of the QVM technology is the creation of a verifiable computing environment that enables users to outsource computations and obtain proofs that the computations were done correctly at random checkpoints. This can lead to the development of a cloud computing market where any user can participate and ensure the trustworthiness of the system. While this system may not be suitable for all tasks, parallelizable projects like SETI@home, folding@home, and genetic algorithms can be easily implemented on top of it. By using spot-checking and security deposits, users can ensure that the nodes cannot cheat for profit.
- Decentralized gambling. The Quantum Smart Chain can facilitate peer-to-peer gambling protocols, providing secure and transparent transactions with minimal fees. Simple protocols like a contract for difference on the next block hash can be implemented, while more sophisticated gambling services can be built on top of the blockchain. By leveraging smart contracts, users can trust that the system is fair and tamper-proof, eliminating the need for intermediaries and reducing costs. This opens up opportunities for new and innovative gambling applications that can be executed in a decentralized, peer-to-peer manner, offering a level of trust and transparency that traditional gambling services cannot match.
- Decentralized prediction markets. With the help of an oracle or SchellingCoin, prediction markets can be effortlessly integrated into Quantum Smart Chain, allowing users to speculate on future events such as election outcomes, stock prices, and sports events. Prediction markets that use SchellingCoin as their data source are incredibly trustworthy and can be leveraged as a governance protocol for decentralized organizations, making it a promising use case for Quantum Smart Chain. This technology can provide a more accurate forecast than traditional polls and surveys, and can incentivize more informed decision-making by rewarding those who make correct predictions.

- Quantum Smart Chain offers on-chain decentralized marketplaces, which are built on a robust identity and reputation system. This system forms the foundation of our marketplace, enabling secure and transparent transactions. With Quantum Smart Chain, buyers and sellers can engage in peer-to-peer transactions without the need for intermediaries, resulting in reduced costs and increased efficiency. Our identity and reputation system ensures that all marketplace participants are authenticated and accountable, promoting trust and facilitating a thriving ecosystem.

IV. Miscellanea And Concerns

1. Modified GHOST Implementation

- The Greedy Heaviest Observed Subtree (GHOST) protocol is a pioneering innovation first introduced by Yonatan Sompolinsky and Aviv Zohar in December 2013, designed to tackle issues arising in blockchains with fast confirmation times. Quantum Smart Chain, a new blockchain, aims to adopt the GHOST protocol to optimize its performance and maintain robust security.
- In blockchains with rapid confirmation times, high stale rates can compromise security. This issue arises because blocks need time to propagate throughout the network. If miner A mines a block and miner B mines another block before receiving miner A's block, miner B's block will be wasted and will not contribute to network security.
- Moreover, there is a centralization concern: if miner A is a mining pool with 30% hash power and miner B has 10% hash power, miner A faces a 70% risk of producing a stale block, while miner B faces a 90% risk. Consequently, if the block interval is short enough for the stale rate to be high, miner A will be substantially more efficient due to its larger size. This situation can lead to one mining pool gaining a significant percentage of network hash power, resulting in de facto control over the mining process.
- The GHOST protocol, as implemented in Quantum Smart Chain, addresses these challenges by modifying the blockchain's structure. This allows for faster confirmation times without sacrificing security or causing centralization. By incorporating "stale" blocks into the decision-making process, GHOST enhances Quantum Smart Chain's overall security and mitigates the centralization risks associated with mining pools of varying hash power. With the integration of GHOST protocol, Quantum Smart Chain aims to deliver a secure, decentralized, and efficient blockchain solution for various applications.
- In the Quantum Smart Chain whitepaper, the GHOST protocol's implementation, as outlined by Sompolinsky and Zohar, addresses the primary issue of network security loss by

incorporating stale blocks in the determination of the "longest" chain. This involves considering not only a block's parent and further ancestors, but also the stale descendants of the block's ancestor (referred to as "uncles" in Quantum terminology) when calculating which block has the greatest total proof-of-work support.

- To tackle the secondary issue of centralization bias, Quantum Smart Chain expands upon Sompolinsky and Zohar's protocol by providing block rewards to stale blocks as well. Under this system, a stale block receives 87.5% of its base reward, while the nephew block, which includes the stale block, earns the remaining 12.5%. It is important to note, however, that transaction fees are not awarded to uncles.
- Quantum Smart Chain adopts a simplified version of GHOST, similar to Ethereum, with a depth limit of seven levels. The implementation is defined as follows:
    a. A block must specify a parent and can indicate 0 or more uncles.
    b. For an uncle to be included in block B, it must meet these criteria:
        1. *It must be a direct child of the kth generation ancestor of B, where 2 <= k <= 7.*
        2. *It cannot be an ancestor of B.*
        3. *An uncle must be a valid block header but is not required to be a previously verified or even valid block.*
        4. *An uncle must be distinct from all uncles included in previous blocks and all other uncles included in the same block (non-double-inclusion).*
    c. For every uncle U in block B, the miner of B receives an additional 3.125% added to its coinbase reward, and the miner of U gets 93.75% of a standard coinbase reward.
- Quantum Smart Chain utilizes a limited version of GHOST, allowing uncles to be included only up to 7 generations, for two primary reasons. Firstly, an unlimited GHOST implementation would introduce excessive complexity into the determination of valid uncles for a specific block. Secondly, an unlimited GHOST with the compensation mechanism employed in Quantum eliminates the incentive for a miner to mine on the main chain instead of an attacker's public chain. By limiting GHOST to a 7-generation depth, Quantum Smart Chain maintains the balance between network security and efficiency while preserving miners' incentives to support the main chain.

2. Fees

- As every transaction added to the blockchain imposes a cost on the network, including the need for downloading and verifying the transaction, a regulatory mechanism, typically

transaction fees, is required to prevent abuse. The default method, employed by Bitcoin, relies on voluntary fees and depends on miners as gatekeepers to establish dynamic minimums. While this market-based approach has been well-received within the Bitcoin community, allowing supply and demand between miners and transaction senders to determine the price, it has its drawbacks.

- Contrary to the intuitive notion of transaction processing as a service provided by miners to senders, in reality, each transaction included by a miner must be processed by every node in the network. Consequently, the majority of transaction processing costs are borne by third parties, not the miner making the inclusion decision. This situation creates a potential tragedy-of-the-commons problem, where individual miners' decisions, based on their self-interest, may lead to negative consequences for the overall network. Quantum Smart Chain aims to address these challenges and ensure a more balanced approach to transaction processing and fee management.
- Interestingly, this market-based mechanism flaw can be mitigated when given a specific inaccurate simplifying assumption. The argument can be broken down as follows. Let's assume that:
    1. A transaction results in k operations, providing a reward of kR to any miner who includes it. Here, R is determined by the sender, and both k and R are (approximately) visible to the miner beforehand.
    2. An operation has a processing cost of C for any node (i.e., all nodes have the same efficiency).
    3. There are N mining nodes, each possessing an equal share of processing power (i.e., 1/N of the total).
    4. No non-mining full nodes are present in the network.
    5. Under these assumptions, the market-based mechanism can balance itself, preventing the tragedy-of-the-commons problem. Quantum Smart Chain takes these considerations into account while designing its fee structure and network management to ensure efficient transaction processing and a fair distribution of costs among network participants.
- A miner will be willing to process a transaction if the expected reward is greater than the cost. Therefore, the expected reward is kR/N, as the miner has a 1/N chance of processing the next block, and the processing cost for the miner is simply kC. As a result, miners will include transactions where $kR/N > kC$, or $R > NC$. It is important to note that R represents

the per-operation fee provided by the sender and serves as a lower bound on the benefit that the sender gains from the transaction. On the other hand, NC signifies the cost to the entire network of processing an operation. Consequently, miners are incentivized to include only those transactions for which the total utilitarian benefit surpasses the cost. In Quantum Smart Chain, this incentive structure ensures that miners prioritize transactions that contribute positively to the network's overall efficiency and security.

- In reality, there are several significant deviations from the assumptions mentioned earlier:
    1. The miner incurs a higher cost to process the transaction compared to other verifying nodes since the additional verification time delays block propagation and increases the chance of the block becoming stale.
    2. Non-mining full nodes do exist.
    3. The distribution of mining power may become highly unequal in practice.
    4. Speculators, political adversaries, and malicious actors whose utility functions include causing harm to the network do exist, and they can create contracts where their cost is significantly lower than the cost incurred by other verifying nodes.

  (1) creates a tendency for the miner to include fewer transactions, and (2) increases NC; as a result, these two effects at least partially cancel each other out. (3) and (4) are the major concerns; to address them, we implement a floating cap: no block can contain more operations than BLK_LIMIT_FACTOR times the long-term exponential moving average. Specifically:

  blk.oplimit = floor((blk.parent.oplimit * (EMAFACTOR - 1) + floor(parent.opcount * BLK_LIMIT_FACTOR)) / EMA_FACTOR)

  For the time being, BLK_LIMIT_FACTOR and EMA_FACTOR are set as constants at 65536 and 1.5, respectively. However, these values are subject to change after further analysis. Quantum Smart Chain considers these deviations and implements the floating cap mechanism to ensure the network's stability, security, and equitable distribution of resources.
- There is another factor that discourages large block sizes, similar to Bitcoin: larger blocks will take longer to propagate and, as a result, have a higher probability of becoming stale. In Quantum Smart Chain, blocks that consume a significant amount of gas can also experience longer propagation times, both because of their larger physical size and the extended time required to process and validate the transaction state transitions. While this delay disincentive is a major consideration in Bitcoin, it is less prominent due to the implementation of the

GHOST protocol. Consequently, relying on regulated block limits in Quantum Smart Chain provides a more stable foundation for maintaining network efficiency and security.

3. Computation And Turing-Completeness

- As detailed in the state transition section, our approach requires a transaction to set a maximum number of computational steps allowed for execution. If the execution surpasses this limit, the transaction is reverted, but fees are still paid. Messages function similarly. The motivation behind our solution can be illustrated with the following examples:
    a. An attacker creates a contract with an infinite loop and sends a transaction to activate the loop to the miner. The miner processes the transaction, runs the infinite loop until it runs out of gas, and stops midway. Despite the halted execution, the transaction remains valid, and the miner claims the fee from the attacker for each computational step.
    b. An attacker designs a lengthy infinite loop intending to occupy the miner's computation resources for an extended period, causing more blocks to be generated before the miner can include the transaction and claim the fee. However, the attacker must provide a STARTGAS value limiting the number of computational steps, allowing the miner to know in advance that the execution will take an excessive amount of steps.
    c. An attacker encounters a contract with code like send(A,contract.storage[A]); contract.storage[A] = 0 and sends a transaction with just enough gas to execute the first step but not the second (i.e., making a withdrawal without decreasing the balance). The contract author doesn't need to protect against such attacks since any changes will be reverted if the execution stops halfway.
    d. A financial contract uses the median of nine proprietary data feeds to minimize risk. An attacker gains control of one data feed, designed to be modifiable via the variable-address-call mechanism as described in the DAO section, and alters it to run an infinite loop. This change attempts to force any claims on funds from the financial contract to run out of gas. However, the financial contract can impose a gas limit on the message to circumvent this issue.
- In Quantum Smart Chain, these examples demonstrate the effectiveness of implementing gas limits in transactions and messages to safeguard against malicious activities and maintain the integrity of the network.

4. Currency And Issuance

- The Quantum Smart Chain network incorporates its native currency, qubit, which serves a dual purpose. First, it provides a primary liquidity layer, enabling efficient exchange between various types of digital assets. Second, and more importantly, it offers a mechanism for paying transaction fees. To ensure convenience and avoid potential denomination debates (similar to the mBTC/uBTC/satoshi debate in Bitcoin), the denominations for Quantum Smart Chain are pre-defined as follows:

  1: atom

  10^{12}: boltz

  10^{15}: farad

  10^{18}: qubit

- This denomination system can be understood as an extended version of the concept of "dollars" and "cents" or "BTC" and "satoshi". In the near future, we anticipate "qubit" to be utilized for standard transactions, "farad" for microtransactions, and "boltz" and "atom" for technical discussions related to fees and protocol implementation. The other denominations might become relevant later on but should not be incorporated into clients at this stage.
- The issuance model for Quantum Smart Chain will be as follows:
  a. Qubit, the native currency, will be released in a token sale at a rate of 1000-2000 qubit per BTC, a mechanism designed to fund the Quantum Smart Chain organization and finance development, similar to successful strategies employed by other platforms such as Mastercoin and NXT. Early buyers will enjoy greater discounts. The BTC obtained from the sale will be exclusively used to pay salaries and bounties to developers and invested in various for-profit and non-profit projects within the Quantum Smart Chain and cryptocurrency ecosystem.
  b. 0.099x the total amount sold (60102216 QUBIT) will be allocated to the organization to compensate early contributors and cover QUBIT-denominated expenses before the genesis block.
  c. 0.099x the total amount sold will be maintained as a long-term reserve.
  d. 0.26x the total amount sold will be allocated to miners each year indefinitely after that point.

5. Long-Term Supply Growth Rate (percent)

- In the Quantum Smart Chain model, the two main choices are (1) the existence and size of an endowment pool, and (2) the existence of a permanently growing linear supply, as opposed to a capped supply like in Bitcoin. The rationale for the endowment pool is as follows:

- If there were no endowment pool, and the linear issuance was reduced to 0.217x to maintain the same inflation rate, the total quantity of qubit would be 16.5% less, making each unit 19.8% more valuable. Consequently, in equilibrium, 19.8% more qubit would be purchased in the sale, rendering each unit's value equivalent to its previous level. The organization would then possess 1.198x more BTC, which can be regarded as two separate portions: the original BTC and an additional 0.198x.
- This scenario is essentially identical to the endowment pool, but with one crucial distinction: the organization holds solely BTC and, therefore, lacks any incentive to support the value of the qubit unit.
- The permanent linear supply growth model addresses some concerns about wealth concentration in cryptocurrencies like Bitcoin and provides individuals in both the present and future an equitable opportunity to acquire currency units. Simultaneously, it maintains a strong incentive to obtain and hold ether, as the "supply growth rate" as a percentage still approaches zero over time.
- Furthermore, it is theorized that since coins are continually lost due to carelessness, death, and other factors, and coin loss can be modeled as a percentage of the total supply per year, the total currency supply in circulation will eventually stabilize at a value equal to the annual issuance divided by the loss rate. For example, with a loss rate of 1%, once the supply reaches 26X, then 0.26X will be mined, and 0.26X will be lost every year, creating a balanced equilibrium. This model ensures a more sustainable and fair distribution of the currency, while still incentivizing participation and retention.
- The note mentions that Quantum is likely to switch to a proof-of-stake (PoS) model for security in the future, which would significantly reduce the issuance requirement to somewhere between zero and 0.05X per year. This change would make the Quantum network more energy-efficient and secure.
- In case the Quantum organization loses funding or ceases to exist for any reason, a "social contract" is proposed: anyone can create a future candidate version of Quantum, with the only condition being that the quantity of ether must be at most equal to 60102216 * (1.198 + 0.26 * n), where n represents the number of years after the genesis block. This condition ensures that the new candidate versions adhere to a pre-defined ether issuance model.
- Creators of the candidate versions are free to crowd-sell or assign some or all of the difference between the PoS-driven supply expansion and the maximum allowable supply expansion to fund development. If candidate upgrades do not comply with the social

contract, they may justifiably be forked into compliant versions, ensuring that the Quantum network continues to operate under the agreed-upon rules and principles.

6. Scalability

- Scalability is a problem with Quantum Smart Chain that many people have. Similar to Bitcoin, Quantum Smart Chain has the drawback of requiring each network node to process each transaction. The Bitcoin blockchain is currently 15 GB in size and is expanding by 1 MB every hour. The Bitcoin network would expand by 1 MB every three seconds if it were to process Visa's 2000 transactions per second (1 GB per hour, 8 TB per year). Quantum Smart Chain is likely to experience a similar growth pattern, which will be made worse by the fact that many applications will be built on top of the Quantum Smart Chain blockchain rather than just a single cryptocurrency as is the case with Bitcoin, but will be made better by the fact that full nodes only need to store the current state of the network rather than the entire blockchain history.
- A big blockchain size has the drawback of increasing the possibility of centralization. Just a very tiny number of very large organizations would likely maintain complete nodes if the blockchain size increased to, say, 100 TB, and all normal users would instead likely use light SPV nodes. This raises the possibility that the complete nodes could come together and decide to cheat in some way that would be profitable (eg. change the block reward, give themselves BTC). Light nodes wouldn't be able to recognize this right away. Of course, there would probably be at least one honest full node, and after a few hours, word of the fraud would start to spread through channels like Reddit, but by then it would be too late. It would be up to the regular users to organize a campaign to blacklist the given blocks—a vast and probably impossible coordination challenge comparable to carrying out a successful 51% attack. There is a problem with Bitcoin right now, but Peter Todd has recommended a change to the blockchain that will fix the problem.
- Quantum Smart Chain will employ two more methods to address this issue in the near future. First, a lower limit on the number of complete nodes is created by the blockchain-based mining algorithms, which need at least every miner to be a full node. But, after processing each transaction, we will, secondly and more significantly, add an intermediate state tree root to the blockchain. As long as there is a single trustworthy verifying node, the centralization issue can be solved even with centralized block validation using a verification protocol. If a miner publishes an invalid block, that block must either be badly formatted, or the state S[n] is incorrect. Since S[0] is known to be correct, there must be some first state S[i] that is

incorrect where S[i-1] is correct. The verifying node would provide the index i, along with a "proof of invalidity" consisting of the subset of Patricia tree nodes needing to process APPLY(S[i-1],TX[i]) -> S[i]. Nodes would be able to use those nodes to run that part of the computation, and see that the S[i] generated does not match the S[i] provided.

- Another, more advanced attack would involve fraudulent miners broadcasting incomplete blocks, rendering it impossible to identify whether or not a block is authentic due to a lack of complete information. A challenge-response protocol is used as a workaround for this problem. Verification nodes issue "challenges" in the form of target transaction indices, and light nodes treat responses as untrusted until another node, such as a miner or another verifier, provides a subset of Patricia nodes as evidence of the block's validity.

V. Conclusion

- With the help of a highly generalized programming language, the Quantum Smart Chain protocol offers advanced features like on-blockchain escrow, withdrawal limitations, financial contracts, gambling markets, and the like. It was initially envisioned as an enhanced version of a cryptocurrency. The existence of a Turing-complete programming language means that arbitrary contracts can theoretically be developed for any transaction type or application, even though the Quantum protocol would not "support" any of the applications directly. Nevertheless, the fact that Quantum Smart Chain protocol goes much beyond just currency is what makes it more intriguing.
- Protocols for decentralized file storage, computation, and prediction markets, among dozens of other similar ideas, have the potential to significantly improve the efficiency of the computational industry and give other peer-to-peer protocols a significant boost by adding an economic layer for the first time. Furthermore, there is a sizable selection of applications that have nothing to do with money.
- The Quantum Smart Chain protocol uses the idea of an arbitrary state transition function to create a platform with special potential; rather than being a closed-ended, single-purpose protocol intended for a specific range of applications in data storage, gaming, or finance, Quantum Smart Chain is open-ended by design, and we believe that it is incredibly well-suited to serving as a foundational layer for a very large number of both financial and non-financial applications.

VI. Notes and Further Reading

1.Notes

- A savvy reader might observe that a Bitcoin address is actually the elliptic curve public key's hash, not the key itself. Yet, referring to the pubkey hash as a public key in and of itself is totally acceptable cryptographic language. This is so because Bitcoin's cryptography can be thought of as a unique digital signature algorithm, in which the public key is the hash of the ECC pubkey, the signature is the concatenation of the ECC pubkey and the ECC signature, and the verification algorithm entails comparing the ECC pubkey in the signature to the ECC pubkey provided as a public key before comparing the ECC signature to the ECC pubkey.
- Technically, the median of the eleven blocks before that.
- Internally, "CHARLIE" and 2 are both numbers, with the latter being represented in big-endian base 256. The range of possible numbers is 0 to 2^256-1.

2. Further Reading

- Like many other open-source, community-driven software initiatives, Quantum Smart Chain has developed since its start. This article is a good place to start if you want to learn about Quantum Smart Chain's most recent advancements and how protocol changes are made.